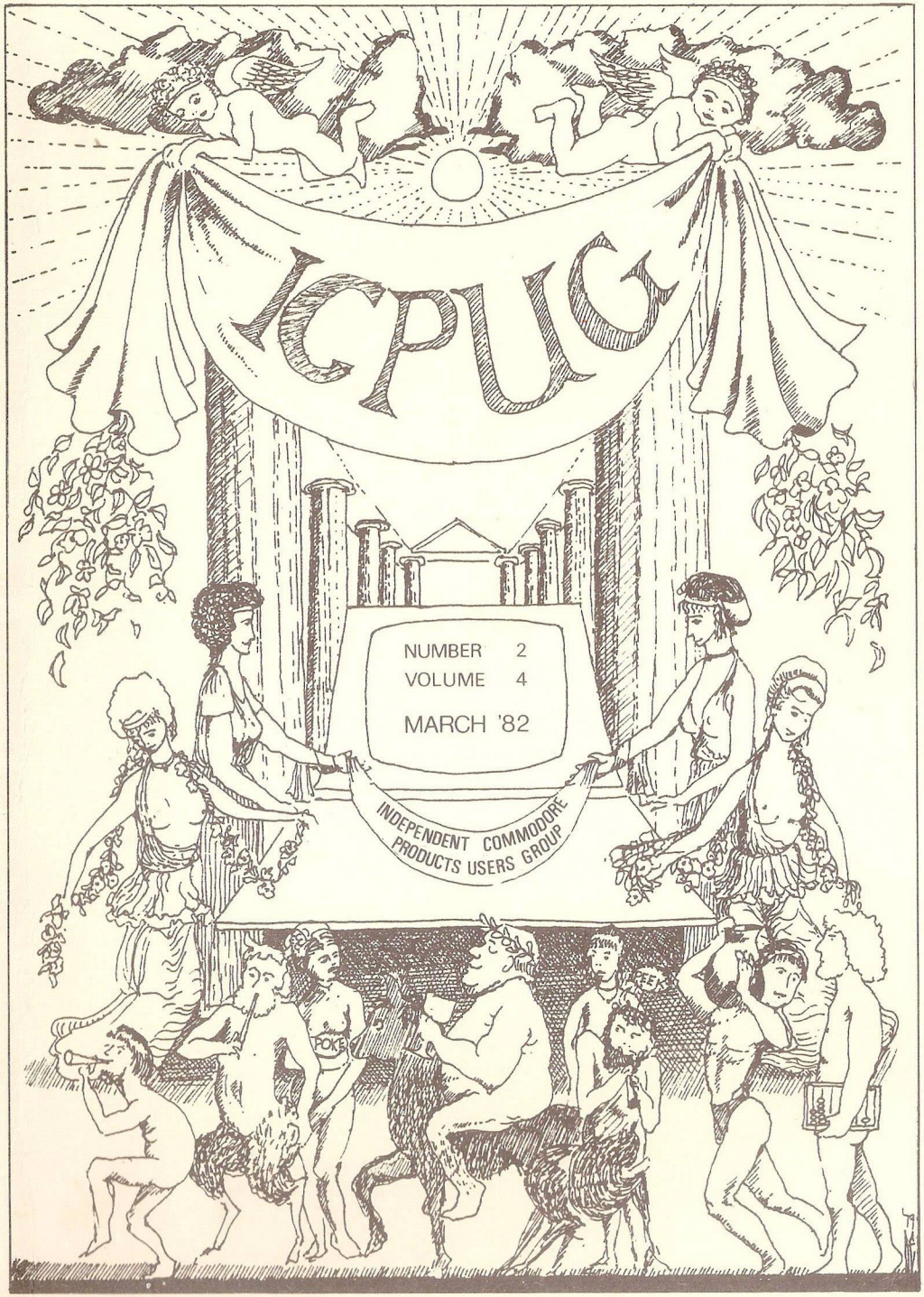

ICPUG
NUMBER 2
VOLUME 4
MARCH '82
INDEPENDENT COMMODORE PRODUCTS USERS GROUP
POKE
PEEK

NATIONAL OFFICIALS

Chairman:	Wing Cdr. Mick Ryan 164 Chesterfield Drive, Riverhead, Sevenoaks, Kent TN13 2EH Telephone: Sevenoaks (0732) 53530
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: (0226) 811585
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

ipug

INDEPENDENT COMMODORE PRODUCTS USERS GROUP

Vol. 4, No. 2 **Newsletter** April 1982

Europe's first independent magazine for PET users

Page	Contents
50	Editor's Notebook
51	PET-Pourri
52	Computer Conference Report
54	DATA Line Writer Program
56	Book Review - Programming the PET/CBM
59	The Fat-40 & the Toolkit
60	Shop Window
61	North London VIC Group & Mini Reviews
63	Slough VIC Group
63	Matters Arising
64	Members' Sales & Wants
66	VIC Matters
78	Disk File
81	Strictly for Beginners
84	Debug
85	The Commodore 4022 Printer
86	Comal Update
88	Magic Squares (Comal)
95	Cassette Merge
96	Commodore Strike Again
99	String Recovery
100	Commodore Column
102	ICPUG Projects

EDITOR'S NOTEBOOK

This time of year is usually a time of change and our User Group is not immune to this. Firstly at the A.G.M. it was agreed to hold a referendum regarding the Group's name. The proposal was that we be known as Independent Commodore Products Users Group (ICPUG). The voting at the closing date was 273 for, and only 96 against. the remainder either didn't vote, didn't mind, didn't know or just didn't. Design of a new logo is in hand.

One of the problems of being General Secretary is that one sometimes receives queries of a technical nature and then has to refer them on which increases considerable the amount of correspondence. Jim Tierney has now offered to assist as General Secretary thus freeing Eli to perform her other tasks as Regional Group Co-ordinator. This division of the work-load has come at a convenient time, since the introduction of the VIC will result in many more local groups. (see elsewhere in this issue).

The user group that was offered by Commodore has proved to be an unacceptable burden and their magazine, editor Pete Gerrard, and two other staff have defected to Nick Hampshire Publications. The group itself is no longer. The editor extends a welcome to all you new members.

Now most of you have heard of 'Printout' magazine and 'VIC Computing', well both of these publications have now been taken over by the Paradox Group, and Dennis Jarrett becomes editor of VIC Computing.

Well that's the changes. Some things do not change; I would still like to hear from regional group members so that the rest of the world knows what you are doing. It need only be a few lines to say 'not a lot', or like one region, producing a cheap modem (details when ready), or another where they are experimenting with networking. Let's hear from you, someone somewhere may already have solved your problem.

R.D.G.

--oOo--

PET-POURRI.

By Nick Higham

(1) While developing a muliple-precision division routine I came across an astounding example of PET's arithmetic foibles. Try the following:

```
PRINT 10^3*944974
944974000
PRINT 10^3*944974+0
944974001
PRINT 0+10^3*944974
944974000
```

The perpetrator of this apparent refutation of the basic law of arithmetic X + 0 = X. is that dreaded beast ROUND-OFF ERROR. This is the name given to the (usually small) error introduced when a number is not stored exactly in a floating point variable. Normally, round-off errors are so small as to be negligible and they often effectively cancel during a calculation.

The above example is unusually extreme and it is the exponentiation operator which has produced these inaccuracies. In general, if accurate results are required, it is expedient to use repeated multiplication instead of exponentiation (where possible).

(2) Interesting effects can be obtained by POKING into REM statements. LOAD or write a short program containing a few REMs, then add the following lines at the start of the program:

```
1 A=42:REM A=124 FOR OLD ROM
2 M=PEEK(A)+256*PEEK(A+1):INPUT C
3 FOR I=1024TO M:IF PEEK(I)=143 THEN POKEI+1,C
  NEXT:LIST
```

These three lines POKE the value C into the first character of each REM statement (code 143). C=1 makes the REMs come out in double width characters on the PET printer, and is a useful way of highlighting blocks of code in a listing. C=18 is of similar use since it produces REMs in reverse field. Other values worth trying are 13 (return) and 17, 19, 20 & 29 (cursor control characters). Possible uses for the latter are to make listings to the screen unintelligable (to say the least)!

--oOo--

COMPUTER CONFERENCE REPORT

By Lawrie Beeching.

The first PET Education Conference was held in London from 27th-29th November 1981. It was attended by delegates from all parts of the UK and Western Europe and supported the interests, not only of schools, colleges and universities, but also of industry and commercial computing. Contributions covered a wide range of topics and revealed much of the forward thinking of educationalists, manufacturers and designers.

On the hardware side, a recurring theme concerned the transmission of software by means of television and telephone. It was clear that a number of schemes involving PRESTEL. CEEFAX and ORACLE were being considered. Commodore's own development, PETNET, anticipates downloading software from a central mainframe to intelligent micros over the telephone at a cost to the receiver of 15 - 20 pence per kilobyte. An even greater interest was shown in the software field. Standardisation of languages was considered and rejected as being unworkable, as it was against commercial interests. COMAL (Common Algorithmic Language), it was hoped, would be capable of being implemented on all machines without the need to introduce any dialects (as now found in BASIC).

Many speakers referred to the storage and distribution of Educational programs and all pointed to the need for this material to be made freely available. It was gratifying to hear so many teacher/programmers who burn much midnight oil developing good programs declare their intention to put this work in the public domain and make it available for others to use without cost. A number of these worthwhile programs were demonstrated and discussed and of the contributions that caught this writer's imagination, the following were of particular interest.

Dr. Christopher Smith modelled a program to forecast a sick patient's progress based on current physiological data. As this program was enhanced, the PET was connected to the monitor of an intensive care unit and set to anticipate the state of a patient in five minutes time.

Thus instead of the alarm bells ringing when a patient's condition departed from preset limits, the staff had five minutes warning of a possible problem.

Adrian Oldknow designed a 'simple' (his word, not mine) number crunching program which allowed the outline of say a house or a crystal to be displayed on the screen. This could then be revolved about any axis, sheared in any direction, stretched, reversed and sectioned. The applications to the teaching of technical drawing, art and design seemed unlimited.

Peter Avis, teaching 14 - 16 year olds, let his students loose in the lab. with meccano, hardboard, plastics, etc., and within a lesson or so had elementary robots coursing the floor, disco lights working and domestic heat balances being modelled and analysed; all connected and being controlled by PETs. No one could doubt his conviction when he demonstrated the interfacing and said, "It's easy, the kids do it all".

Peter Bishop analysed all the A-level computer syllabuses and designed the 'virtual computer' which suited each examination board's requirements, but had none of the sales features which are peculiar to the commercial micros. He then designed a program which made all the micros behave like his virtual model. So now his students not only study the architecture, but are able to implement their own programs without having to worry about the constraints imposed by particular manufacturers. His work overcomes the problem of lack of standards and is so good that he has 'gone commercial'. However, at ##50 for a comprehensive package he can hardly be accused of extortion.

Many other interesting projects were described and are currently available through software libraries and workshops. Trevor Lusty described his development of administrative programs for school records, timetabling, etc. Keith Tomlinson spoke of the work done in the City of Bradford where the support system for computers in schools is total. The B.B.C. were criticised for basing their forthcoming course on a particular micro. However, the series of programmes was eagerly awaited.

--oOo--

PROGRAM NAME: DATALINE INST

```
 10 PRINT"<clr><rvs>DATA LINE WRITER<off>:- CONVERTS A
    SEQUENTIAL FILE ON TAPE INTO DATA";
 20 PRINT" LINES WHICH CAN BE APPENDED TO A BASIC PROGRAM.
    IT WORKS ON";
 30 PRINT" BASIC 2 OR 4.":PRINT
 40 PRINT"STRINGS ARE INPUT, CARRIAGE RETURNS ACTING AS
    SEPERATORS, AND";
 50 PRINT" CONVERTED INTO A PROGRAM LINE. (LINES ARE BUILT
    UP TO 80 CHARACTERS.)
 60 PRINTPRINT"<rvs>TO USE<off>:LOAD DATA LINE WRITER.":
 70 PRINT"<dn>SET UP FILE TO BE READ AND TYPE RUN.<dn>"
 80 PRINT"DATA LINE WRITER WILL READ THE FILE, CONVERT IT
    AND POKE IT INTO";
 90 PRINT" HIGHER MEMORY. WHEN AN END OF FILE MARK IS
    REACHED A";
100 PRINT" MACHINE CODE ROUTINE IS AUTOMATICALLY RUN ON
    THE SCREEN TO MOVE";
110 PRINT" THE DATA STATEMENTS TO THE BASIC PROGRAM AREA."
120 GOSUB1000
130 PRINT"<clr>THE BASIC POINTERS ARE THEN RESET."
140 PRINT"WHEN THE PET RETURNS WITH THE READY MESSAGE
    CLEAR THE SCREEN";
150 PRINT" YOU CAN THEN LIST AND SAVE THE PROGRAM."
160 PRINT"(N.B. STRINGS OF 69 TO 248 CHARACTERS WILL BE
    WRITTEN AS ONE ";
170 PRINT"LINE. CARRIAGE RETURNS THROUGH THESE LINES WILL
     LOSE THE EXCESS";
180 PRINT" CHARACTERS !)"
190 PRINT"<dn>ON APPENDING TO YOUR BASIC PROGRAM CONVERT
    INPUT# AND GET# ";
200 PRINT"ROUTINES TO APPROPRIATE READ STATEMENTS.<dn>"
210 PRINT"IF DATA LINE WRITER FINDS INSUFFICIENT  MEMORY
    OR A STRING LONGER";
220 PRINT" THAN 248 CHARACTERS THE PROGRAM TERMINATES."
230 GOSUB1000:"<2dn>LOAD DATA LINE WRITER WHEN READY.":END
1000 PRINT"<rvs><12sp>PRESS ANY KEY<16sp><off>
1005 GETA$:IFA$<>""THEN1005
1010 GETA$:IFA$=""THEN1010
1015 RETURN
```

PROGRAM NAME: DATALINE WRITER

```
100 PRINTCHR$(147);"DATA LINE WRITER:BY:-P.N.MORTIBOY"
110 LL=68:I=10:LN=10000:OP=1025:CP=3524:MS=PEEK(52)+256
    *PEEK(53):LC=128:QU=2
120 POKE53,13:POKE52,196:POKE49,13:POKE48,196:OPEN1,1,0:
    GOSUB310
130 GET#1,A$:IFA$=""THENA$=CHR$(0)
140 IFASC(A$)>LCORA$=CHR$(44)THENQ=QU
150 IFA$<>CHR$(13)THENB$=B$+A$:GOTO130
160 IFQ=QUTHENB$=CHR$(34)+B$+CHR$(34)
170 IFLEN(B$)>248THENPRINT"STRING TOO LONG":GOTO290
180 IFLEN(C$)+LEN(B$)>LLTHENGOSUB220:GOSUB300:C$=""
190 C$=C$+B$+",":K=1:B$=""
200 IFST=64THENGOSUB220:GOTO250
210 Q=0:GOTO130
220 L=LEN(C$):IFCP+T+L>MSTHENPRINT"INSUFFICIENT MEMORY":
    GOTO290
230 FORJ=1TOL-K:X=ASC(MID$(C$,J,1)):GOSUB340:NEXT:K=0
240 PRINTLN;"DATA ";LEFT$(C$,L-1):LN=LN+I:RETURN
250 X=0:GOSUB340:GOSUB360:GOSUB340:GOSUB340
260 PRINTCHR$(147);:FORI=33008TO33061:READX:POKEI,X:
    NEXT
270 NU=CP+T+1:GOSUB350:POKE33062,LO:POKE33063,HI
280 PRINT"SYS33008:CLR":POKE623,19:POKE624,13:POKE158,2
290 CLOSE1:NU=MS:GOSUB350:POKE53,HI:POKE49,HI:POKE48,LO:
    POKE52,LO:END
300 X=0:GOSUB340:GOSUB360
310 NL=CP+T:T=T+QU:NU=LN:GOSUB350
320 X=LO:GOSUB340:X=HI:GOSUB340
330 X=131:GOSUB340:X=32:GOSUB340:RETURN
340 POKECP+T,X:T=T+1:RETURN
350 HI=INT(NU/256):LO=NU-HI*256:RETURN
360 NU=OP+T:GOSUB350:POKENL,LO:POKENL+1,HI:RETURN
370 DATA173,196,13,141,1,4,238,241,128,208,3,238,242,
    128,173,39,129,205,242,128
380 DATA 208,10,173,38,129,205,241,128,208,2,240,11,238,
    244,128,208,3,238,245
390 DATA128,76,240,128,173,244,128,133,42,173,245,128,
    133,43,96
```

--oOo--

BOOK REVIEW

Programming the PET/CBM
By Raeto West
£ 14.90

Level Ltd.,
PO Box 438
Hampstead,
London NW3 1BH

This publication represents over a year's intensive research by one of our members and the resulting product is a valuable work of reference. A tremendous amount of useful information has been packed into this 500+ page work at which I was so over-awed that I did not know how to start this review at first. So let me start at the beginning with a conducted tour through the book.

The 'contents' runs to so many pages that there is a contents page for the contents ! Chapter 1 is the introduction and overview and discusses the necessary limitations of the book (eventually one has to stop writing and publish something). The VIC, 9000-series and modems receive little or no coverage so that the author could concentrate on the more common configurations. The introduction concludes with a chronology of Commodore microcomputers.

Chapter 2 covers BASIC and how it works (no it's not a rip-off of Mike Todd's article). Description is by example and includes a section on optimising BASIC and ROM differences (although these are discussed where relevant throughout the text).

Program and system design are covered on a short chapter, followed by 'Effective programming in BASIC', a series of useful hints, tips & techniques. An example is a parameterised crash-proof input routine. Searching, sorting and string manipulation are discussed, although somewhat briefly in comparison with the depth of coverage of other items in the book.

Chapter 5 covers BASIC keywords, but the list has been extended to cover extensions by common utilities and some not so common, plus other BASICs. In my opinion they have no relevence in the book, although the author does attempt to give routines to perform such tasks as HTAB, VTAB,

INSTRING$. MERGE and these would be more appropriate to a different chapter. Each keyword is examined and explained at length and I mean at length. for example the PRINT statement includes a flow-chart of the processing of this function and the description runs to three pages.

Chapter 6 concerns disk drives. It contains so much useful information on disks that the forth-coming ICPUG publication on disk drives has been abandoned. I will not go into detail suffice to say that nowhere else has so much useful information been published about the workings and techniques of using the Commodore disk drives. Chapter 7 continues the theme covering the disk commands in the same format as the chapter on BASIC keywords and not only covers differences due to versions of BASIC & DOS, but also DOS Support. I don't agree with the author that SCRATCH is a word peculiar to Commodore, it is used on DEC systems, to name but one.

Chapter 8 covers 'other peripherals and hardware'. The C2N cassette deck is described as having a short cable (as has my own) but some models have a generous length of lead as a point of contention. The meat of the chapter is devoted to a detailed treatise on cassette operation and 4022 printer. The keyboard is regarded by the author as a peripheral and more written about it than I would have thought possible.

Graphics & Sound are covered in chapter 9 and new material is represented by a discussion on the CRT controller chip used in 12" screen models. Throughout the book there have been obvious problems in getting satisfactory reproduction of the PET graphic characters although the results obtained are sufficient. Animation and plotting are covered before the chapter moves on to sound generation. CB2 sound and 8-bit sound is covered with some examples.

Chapter 10 is an introduction to machine code programming and includes a description of the TIM monitor as modified for use in the CBM models. Supermon, Extramon and others are described. Machine-code programming is

explained the way I like, by example. The following chapter continues the theme on programming the 6502 and includes some useful hints and programming tips and chapter 12 gives an alphabetic reference to the 6502 op-codes in the style of the earlier chapters on keywords and commands.

Chapter 13 discusses the way some of the ROM routines can be used with your machine code and 14 goes on to assemblers and to an explanation of the CHRGET routine, wedges, utilities and debugging. Strangely the same chapter wanders into the IEEE-488 bus, VIA & PIA chips.

Chapter 15 are similar to the ROM tables in the IPUG Compendium, but cover BASIC4 and have more detail. I would buy the book for this chapter alone were it not for the forth-coming ICPUG ROM Gazetteer.

Chapter 16 covers mathematical programming, statistics, simulation, accounting, trigonometry, array and matrices, number theory and curve fitting. The chapter concludes with the mathematics routines in BASIC ROM.

The final chapter covers programming for business and education, although no examples are given, simply a discussion with some interesting quotes at each subsection.

The book is completed with appendices and a lengthy index. From the above you will see that there is not just 'something for everyone' but 'plenty for everyone'. The author has aimed at accuracy of information (which will blow his chances of it getting Commodore's Approval !), but I note that there are a few where the original material was in error and the error has been perpetuated. The account of the IEEE-488 EOI in relation to scrolling is not strictly accurate to give another example. This in no way detracts from the usefulness of the book. If you thought that the 'PET/CBM Personal Computer Guide' was good, well this is better, and as for the 'Hitch-Hiker's whatsit', forget it. This book is a must for every CBM/PET user.

R.D.G.

--oOo--

THE FAT 40 AND TOOLKIT.

By Robin Harvey

The introduction of the latest 4000 Series computers with the larger screens (sometimes called THE FAT 40), has shown up a minor problem when TOOLKIT is used. The problem shows up when the TRACE function of TOOLKIT is used. Now the line numbers that are displayed at the top right hand corner of the screen, become double spaced.

The reason for this is that TOOLKIT for BASIC 4.0 is able to distinguish whether it is installed in a 4000 or 8000 series machine and modify the line number display accordingly. Unfortunately this mechanism does not work with the FAT 40 series and TOOLKIT thinks it is in an 8000 series machine. TOOLKIT has in its program a machine code instruction at address $A53E of LDA $E000 which loads the content of address $E000 into the Accumulator and deduces from that the machine type. In the old 4000 series this address held $A9 but in the FAT 40 this is now $4C because the $E000-$E7FF ROM has been drastically changed in the new machine.

If one is really worried about this and you have EPROM blowing facilities, it is possible to restore TOOLKIT to its original format with the FAT 40. Another IPUG Member, David Jowett has shown that changing just one byte is enough when blowing a new EPROM. Simply change byte $A53F from $00 to $4B. Line $A53E then becomes LDA $E04B and your TOOLKIT will now correctly work in both types of the 4000 series as well as the 8000 series.

--oOo--

SHOP WINDOW

If you are a business user, you would most likely be rather dismayed if the power failed and you lost data or were unable to complete a transaction as a consequence. What is needed is a no-break power supply such as the 'Power Bank'. Ratings 120, 250, 500 & 1000VA are in production. Prices and details from Power Testing Ltd., 1, St. Marys Lane, Upminster, Essex, RM14 3PA. Tel: Upminster (04022) 26938.

From the people that brought you the Instant ROM (p81 July '81) comes the GCC1 PETCLOCK. This plugs into the user port and is powered by a lithium battery (compare the similar product from Microscience p28) having a battery life of 10 years. Time is available as a 23:59:59 format and date in US or UK numeric formats. CA1 line interrupt rate is a link-selectable option. This is a Commodore Approved product and is available from Greenwich Instruments Ltd., 22, Bardsley Lane, Greenwich, London, SE10 9RF. Tel: 01-853 0868.

If you are serious about using the IEEE-488 bus and find the Commodore implementation inadequate, there is a plug-in ROM containing an IEEE-488 operating system. The 4K program called BOS gives an additional 56 commands, for example BOS provides a monitor that allows you to see the bus data and command bytes on the screen during transfers; several listeners and talkers can be active at the same time; and any byte can be sent both in the IEEE-488 command mode and in the data mode. Two versions are available, Red BOS is for 2000/3000 series, Orange BOS is for 4000 series. ROMs occupy $A000-$AFFF address range. Available from Rhombus, 87, Bourne Way, Hayes, Kent, BR2 7EX.

CIL have come up with a user-port interface giving 4 x 12-bit analogue input, 4 x 12-bit analogue output, 4 x TTL inputs and 4 x relay outputs all on a board with program in EPROM for £ 195. The board connects to both user and cassette ports, but contains a replica cassette connector. A suite of applications programs in EPROM will be available soon. Contact CIL Microsystems Ltd., Decoy Road, Worthing, West Sussex. Tel: (0903) 210474.

The new Supersoft catalogue is simply stuffed with goodies and if you are not on their mailing list, tut, tut. Plug-in chips are a specialty. There is an impressive array of utilities, but how about a word-processor on a chip, or a faster BASIC, or a dual option of either 40-column BASIC4, or Space Invaders in ROM ! Accessories, extensions, add-ons and add-ins are all represented, together with IEEE cables, connectors and zero-insertion-force sockets. Write to Supersoft, First Floor, 10-14, Canning Road, Wealdstone, Harrow, Middlesex, HA3 7SJ. Tel: 01-861 1166.

R.D.G.

--oOo--

NORTH LONDON VIC GROUP

By Jim Chambers

The group now numbers around nine after a four-week start and includes two radio amateurs, G3RDG and myself G4IBK, making use of the VIC for morse and RTTY (teletype) reception. Meetings are held on alternate weeks in members homes so far. Equipment includes two printers and one Arfon motherboard. For more information phone Jim Chambers on 01-387 7050 (University College - daytime).

Brief review of the Arfon Unit:

A high quality metal tray is a good fit on the VIC and matches well in colour and style. It contains various windows for power and ports, and has a clip to hold the TV modulator. The on-board heavy-duty power supply has a toroidal mains transformer with outputs of a) 9 volts ac for the VIC, b) 5 volts regulated for the seven sockets on the printed board to take extra RAM or ROM, c) 24 volts ac, use undisclosed (hint of a cheap printer ?). A metal lid or tray is an option.

One minor irritation is the fact that the 9 volts is fed to the user port, so that if one wants to use the PA0 to PA7 lines, then wires have to be taken to the same plug. There has been some confusion regarding the correct POKE number for this port - it is POKE37138,1 for PA0, and ,2 ,4 ,8 etc for each line. Comma3 for lines 1 & 2 together. All lines sit at 5-volts until POKEd down. Use ,0 to reset. If

you use the eight lines to detect switch closures, then PRINT PEEK(37008) returns a unique number depending on the number of lines switched to ground.

Regarding the joystick, if you are making your own connection note that the FIRE switch goes to pin 6 (marked light pen in the manual), the North switch to pin 1, South to pin 2, East 4, West 3, common to pin 8, all on the 9-way Cannon socket. Some British software games do not use or give you the option of use of the joystick - I find it is vital if you do not want your children to pound the keys. The Creative Software and Audiogenics tapes are OK, Rabbit (Cream) tapes that I have, do not have this option.

Brief review of the Commodore VIC printer:

A very small smart machine, 172 D x 328 W x 132 H in mm. Weight is 2.5Kg. It plugs straight into the VIC without interfaces. A 5 x 7 dot matrix is used by the Uni-hammer going left to right at 30 characters/sec, 80-columns wide. One can have upper/lower case, numerals and PET/VIC graphic symbols. Every dot is addressable should you want to do your own logo. As supplied, the machine takes 8-inch paper which is more difficult to obtain than 8+1/2", and is tractor only, i.e. sprocket holes.

I have modified my machine to friction feed plain paper, i.e A4. I can supply details if you are a model engineering type, otherwise I may be able to supply a kit of parts for same. This mod. does not involve any cutting or alterations that another member has done to make the machine take 8+1/2" tractor feed paper.

All in all it is a nice little machine with 12 special commands including a nice large character set. One detail spoils the high quality print and that is the lack of lower case descenders.

--o0o--

SLOUGH VIC GROUP - MARCH 29TH

The inaugural meeting of the Slough region VIC group will be held at the Slough College of Higher Education at 6.30 p.m. for a 7.00 p.m. start. The proceedings will conclude (or adjourn to the local) at 9.15 p.m. An attraction of the first meeting will be a demonstration by the authors of "Learn Computer Programming with the Commodore VIC", L.R.Carter and E.Huzan. It is hoped that, as well as three VICs being available, the new VIC disk and printer will be on display. Also available will be the PET version of the classic of all computer games, Adventure, plus numerous other games and utilities. One or two control systems developed within the college will also be on display. During the evening there will be some discussion time as to what future meetings might take. Possibilities include software exchange and joint software development, speakers and demonstrations. (With Commodore just down the road, we could have them popping in to show us the latest products and developments), social evenings or outings with a micro connection, and various other forms of information interchange, either formally or informally.

For further details contact Brian Jones, Department of Maths & Computing, Slough College of Higher Education, Wellington Street, Slough. Tel: 34585 Extn 81.

The college is opposite the BR station and bus station at the centre of Slough. This is all at the junction of the A4 and B416. Parking is available in the college grounds.

To give an indication of likely numbers, please contact the organiser.

--o0o--

MATTERS ARISING

Omitted from the review of the 'Wideband Speakeasy' (p4) was the address of Intelligent Artefacts, Cambridge Road, Orwell, Royston, Herts. Tel: Cambridge (0223) 207689.

--o0o--

MEMBERS SALES & WANTS

For sale: C2N cassette unit, practically unused, £ 30.00. [Also compatible with VIC-20 - Ed].

PET software (old ROM) plotter, address book, link, portfolio management - £ 10.00 the lot.

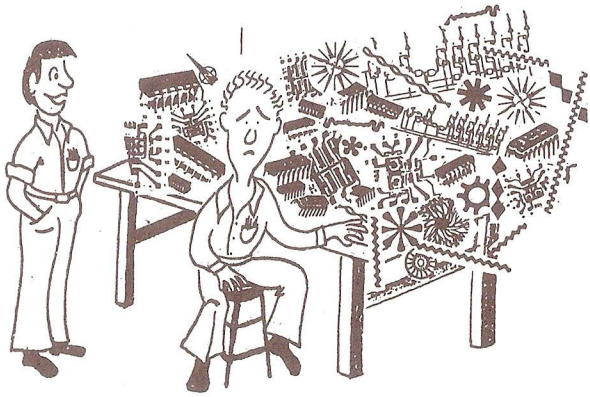
SWTP-40 printer and software, fully documented £ 100.00. D. Miln, tel: 01-446 1877.

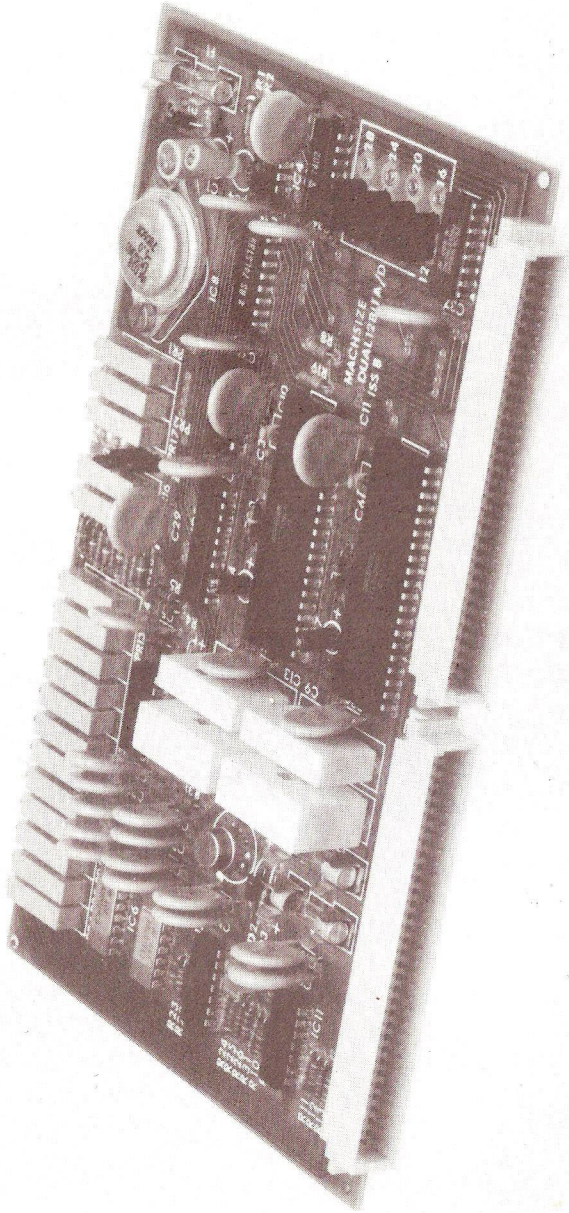
--o0o--

*** PHOTO ***

The new 12-bit A-D converter from Machsize. Also new is a 24-channel multiplexer. Both these interfaces are additional to the range mentioned in Shop Window (pp109 & 113). For details contact Duncan Smyth at Machsize Ltd., tel: Leamington Spa (0926) 312542 & 32399.

--o0o--




MACHINIZE
DUALTRON I/O
CHI ISS 8

VIC MATTERS

By Mike Todd

If you watched Tomorrow's World on February 11th you will have actually seen a VIC doing something reasonably useful. In fact it was keeping a track of cows and their milk production etc. down on the farm. It actually identifies the cows by automatically reading an electronic identification tag attached to the cow's ear. For a programme which has had a preoccupation with the APPLE for some time it is quite something to see a VIC. It also appeared in the first of the BBC Computer programme.

Since my last column I have received several letters, and VIC interest in the Group is beginning to take off. We also have several new ICPUG members who are specifically interested in the VIC. To these I extend a warm welcome and remind them that we are here to help. We will try to answer technical queries (or at least pass them on to someone who can) and to provide a forum for ideas (hardware and software) for using the VIC. All contributions to this column (or even complete articles) are welcome and I look forward to hearing from some of you. Probably one of the most useful forms of contact would be to let us know what you are doing with the VIC and what you think of any of the add-on goodies, be they disk units, printers or games. They don't have to be fully-fledged articles, broad comment is equally useful.

Some of the new blood will have joined us as a result of reading my column in VIC Computing and they may have noticed a distinct similarity between what they have read there, and my column in the Newsletter. In fact VIC Computing asked if they could reprint the ICPUG VIC column, they were told that they could, so of course the columns are the same. In the future the two columns may take on a different emphasis but in the meantime I apologise if you have to put up with me twice...

If anyone wants to start a regional VIC users group there are several things that ICPUG can do to help. If there is an existing ICPUG regional group in the area, then we can put VIC users in touch with them and hope that they will form a VIC "sub-group". If there is no regional group nearby, we can at least give some publicity for the group as well as putting people in touch with others in their area. In addition, ICPUG has a continuing offer from Commodore which will allow regional VIC groups to reclaim some of the initial expenses incurred in setting up the group (up to 50 pounds!). Details of all this can be obtained through Eli Pamphlett (see inside front cover for her address).

Elsewhere in the Newsletter you will see an article by Jim Chambers about the VIC. Jim lives and works in North London and is interested in getting VIC users in that area together. He can be contacted on 01-387-7050. In that area are several VIC/PET users who are currently developing a variety of VIC goodies including an amateur radio package (morse and teletype decoding) - I hope to have more details next time.

Despite efforts to obtain some of the wide range of VIC hardware and software I'm afraid nothing has been forthcoming, however I have managed to try out a couple of the games cartridges. These games cartridges plug into the expansion socket on the back of the VIC and contain one small printed circuit board, one ROM chip and a capacitor! For 18 pounds these represent dubious value bearing in mind most cassette based games are in the 7 to 10 pound bracket and are probably just as good.

The two games that I looked at were AVENGER (which is VIC's answer to Space Invaders), and SUPER LANDER (a graphical version of the simple Moon Lander game). Taking AVENGER first, I was immensely impressed with the graphics and the feel of the game. I didn't play for more than five minutes so cannot comment on the long term excitment. However general opinion is that it is a

huge improvement on the PET implementation, significantly better than the arcade version and faster and more skilful than the Atari version. I would think that this may be worth having if you're a games nut !

SUPER LANDER can only be described as a bit boring. You control left/right and up/down motion of a small spacecraft using the keyboard, trying to land on one of three landing sites. The landing is shown in profile and you have to manoeuvre down between mountains. One interesting feature is that as you get close to the landing site the display shows the descent in close-up. As a cassette game at 6 pounds or so it would be fair value, but 18 pounds - well I wouldn't buy it !

Turning to VIC news - there are a lot of companies springing up dealing in VIC goodies. A large range of games is now available, as well as simple utilities at fairly reasonable prices. I would suggest that there are two principle sources of adverts pointing towards these dealers. The first is VIC Computing, which is devoted to VIC matters but is quite expensive at 95p for a fairly slim volume. This problem is likely to improve as the magazine gathers momentum although only two issues have been seen so far. The second is COMPUTER & VIDEO GAMES which seems to have a number of ads from VIC dealers as well as having a good range of games program listings in BASIC which in many cases could be transferred to the VIC.

The VIC disk drive (which incidentally is reported to be fully compatible with 4040 PET disks) is now available (and was also seen in the background on Tomorrow's World) as is the printer. However, Arfon Micro (in Gwynedd, North Wales) are developing a co-ordinated range of VIC peripherals which start with their VIC-20 expansion system incorporating a very good power supply and slots for seven cartridges on the expansion port. They also have a low cost printer which is due out any day now.

I very much hope that by next issue I will be able to report more fully on the Arfon unit and on some of the hardware add-ons such as the IEEE-488 interface - is it going to be the answer to full PET/VIC peripheral integration ?

Anyway, enough of this, let's get down to some simple arithmetic. Many of you will have read that the VIC runs a 6502A chip which can run at 2MHz - i.e. twice as fast as the PET. Some have automatically assumed that this means that the VIC actually does run twice as fast as the PET. Well, it doesn't. Let us see why.

At the heart of VICs and PETs is a crystal which provides clock pulses to allow the microprocessor, peripheral chips and the video to work correctly. On the PET this is usually 16MHz, which is divided by 16 to get 1,000,000 clock pulses per second. This 1MHz clock is further divided to provide line and frame timing for the video circuitry. the latter is used as the 60Hz interrupt (for the TI clock).

On the VIC, things are slightly different. The main constraint is that the VIC chip must have a clock input of 4.433618 MHz which is the frequency of the colour subcarrier in the British (PAL) TV system. If this were not accurate, TV sets just would not be able to decode the colour information. This frequency is obtained by using a 8.867236 MHz crystal and dividing it by two.

The VIC chip does all the necessary dividing from this clock to provide all the video signals. It also divides it by 4 to produce a 1.1084045 MHz clock for the rest of the VIC. Now it doesn't take much to realise that this clock is 11% (or more accurately 10.84045%) faster than the PET's clock and this is how much faster the VIC will run. The 6502A was chosen because the standard 6502 would be unreliable at this higher clock frequency.

This different clock system leads to two interesting features of the VIC. Firstly, the 60Hz interrupt is no longer derived from the video frame pulses. Instead, one of the 6522 peripheral chips is used to divide the clock by 18470. This produces a TI clock which updates at a frequency of 1108404.5/18470 = 60.011072 Hz. This means that the TI clock updates slightly more than 60 times a second and will in fact run .01845% fast. This means that the TI$ clock will be one second fast after 90 minutes 19 seconds.

There is one interesting feature of this method of obtaining the 60Hz interrupt and that is that the division ratio (initially set to 18470) can be changed very simply by two POKE commands (POKE 37156,LO:POKE 37157,HI) which change the division ratio. For instance if HI=43 and LO=76 then the division ratio is 43*256+76=11084 which results in an interrupt every 1/100 second. Although TI$ will no longer be correct, TI will now increment in hundredths of a second with an accuracy of 0.00041%. Of course the keyboard repeat and the cursor flash rate will be faster too. In fact you can try different values for HI (the lower the value, the faster the interrupt) and make the cursor flash and repeat extremely fast !

Note that any use of cassette input/output routines (which use this timer) will restore the original 60Hz interrupt when they finish.

Cassette tape relies on the 1.11MHz master clock for its timings. Unfortunately the VIC uses exactly the same timing constants as the PET, and so VIC cassettes actually write 11% faster that the PET. This can be heard as a difference in pitch between PET and VIC tapes. The tolerance of the cassette read routines in both VIC and PET is such that this difference can be compensated for, but could be the cause of some read problems that I've had reported.

Staying with cassette operations, it has been pointed out that cassettes SAVEd on old ROM PETs (ie with BASIC1) will load okay on the VIC, except that the first line of the program is corrupted. This doesn't occur with tapes recorded on later PETs. The reason is very simple, but a bit difficult to explain, but here goes.

When you type SAVE, the PET or VIC writes the current program from its first byte to its last. The first byte of the program is actually at location 1024, but PET and VIC actually ignore this first byte (which is always zero anyway) and SAVE from 1025 upwards. However, very early PETs (with BASIC1) actually save this byte as well. So BASIC1 tapes start with an extra zero byte. This does not cause problems on the PET since it always reloads starting at the same location it was SAVEd from so that BASIC1 actually loads back starting at 1024 while other BASICs load from 1025.

This is exactly what the machine wants to see and so there is no problem. The VIC, on the other hand, normally takes no notice of the save address. Instead it loads at one byte above the start of program space - in the case of VICs with only 3.5K add on RAM, this would be 1025. Other VICs would load from 4097 or 4609.

BASIC1 PETs have added an extra zero byte at the start. This shifts the whole program up 1 byte in effect and the line number of the first line is corrupted. There is really no simple way round the problem, (although VICs with only 3.5K expansion - i.e. those in the middle group of the memory maps shown on page 22 of the January Newsletter - can use the absolute LOAD mode on the VIC which will force the program to start loading at 1024 and not 1025.

Still on the subject of LOAD, the last column attempted to show the difference between absolute and relocating LOAD and SAVE. Unfortunately, somehow or other I managed to get some of the information the wrong way round. The following is an attempt to put matters right. An "absolute" LOAD is one which will LOAD back at exactly the same place that it was SAVEd from and would be used for machine code programs which need to be at the same place every time. A "relocating" LOAD will automatically start LOADing the program at the start of the VIC's program space, which could be one of three places (4097, 1025 or 4609 - $1001, $0401 or $1201). The format for LOAD and SAVE is "filename",device,mode.

SAVE mode=0 - header code = 01 - allows relocating LOAD
SAVE mode=1 - header code = 03 - forces an absolute LOAD

LOAD mode=0 - header code = 01 - do relocating LOAD
LOAD mode=0 - header code = 03 - do absolute LOAD

LOAD mode=1 - header code = 01 - do absolute LOAD
LOAD mode=1 - header code = 03 - do absolute LOAD

Note that both modes default to 0. Therefore a SAVE with unspecified mode produces a program with header code 01 and a straight LOAD will relocate the program on loading. Note that the SAVE modes are only operative on cassette tapes, but the LOAD modes will work on non-cassette LOADs as if they had a header code of 01.

Any tapes SAVEd on the PET will have a header code of 01 and will therefore relocate. Also, the PET does not recognise the 03 header code so that SAVEd programs in mode 1 will not load on the PET.

I do hope that this has cleared up the confusion I managed to create in the last column.

It may be worth pointing out at this stage that the VIC only allows LOAD and SAVE to device 01 (cassette) and devices >3 (disks etc). It is not possible to LOAD or SAVE with an RS232 device.

Before going on, it is worth mentioning a bug in the INPUT routine which can cause a whole line of characters to be returned instead of just those characters typed by the user. If you print characters beyond the end of a line, the VIC automatically puts the excess on the next line. It also makes a note of the fact that the new line is effectively a part of the previous line. It is this wrap-around feature which allows a single line of BASIC program text to actually consist of up to four screen lines. Unfortunately, the VIC fails to manage this correctly which results in any INPUT statement on a wrap-around line returning the entire contents of the line instead of the users INPUT.

This can occur if the INPUT prompt string takes you over the end of a line or you go for input on a previously "wrapped-around" line. It does not occur on the first line of a wrap-around line.

I'm afraid that there is no easy solution other than making sure that you do not allow INPUT on a wrap-around line. The problem is very similar to one which occurs on the PET when INPUT is taken from the bottom line of the screen and which is described on page 122 (Sept '81 Newsletter).

Now to a look at the peripheral chips - there are three of them, all in page $90. The first is the VIC chip itself which I don't intend to look at in detail here. The other two are 6522 VIAs (standing for Versatile Interface Adapters). Each has 16 control/data registers, 2 timers, 2 input/output ports, a shift register and sundry interrupt and control ports. I can't possibly explain the detailed operation of these chips here, but I have given details of the chip layout for those who want to experiment further.

The first 6522 (VIA#1) starts at $9110 (37136) and

contains the RS232 port (also used as the user port) on register B, with register A having lots of bits and pieces. The interrupt from VIA#1 is fed to the non maskable interrupt (NMI) which is used for the RESTORE key and RS232 control.

The two timers are used by the RS232 software, the first is the baud rate timer for RS232 transmit and the second is for RS232 receive. In fact what happens is that the RS232 data input line generates an interrupt as soon as a start pulse is received, thereafter it is the T2 timer interrupt which generates the interrupt. On each of these interrupts the RS232 data line is sampled and a complete RS232 byte assembled. Once a full byte has been received, the T2 is disabled and the VIC awaits the next start pulse. This is why the RS232 data input line appears in two places, once on the CB1 line (for start pulse detection) and the other on b0 of register B (to receive the data bit). The RESTORE key is also actioned via NMI.

VIA#2 (starting at $9120 - 37152) has the keyboard row and column (register A and B) lines available (with two of the column output lines doubling as joystick and cassette lines). The VIA#2 interrupt line is fed to the normal interrupt line (IRQ) and is used for cassette input/output timing (using T2) and the normal 60Hz interrupt is generated by T1 timing out.

The full layout is shown in the following two diagrams. Note that in register C in both VIAs the 3-bit controls (for CA2 and CB2) are set to 110 for the line to be set to 0, and 111 for a 1. Also, neither of the shift registers are actually used.

Finally, I mentioned in the last column that the SYS command allows the registers to be passed to it. There are many possible applications of this, but here is a simple one. Some BASICs allow cursor positioning at X,Y on the screen. You can do this on the VIC with:

```
POKE 781,line : POKE 782,position : SYS 65520
```

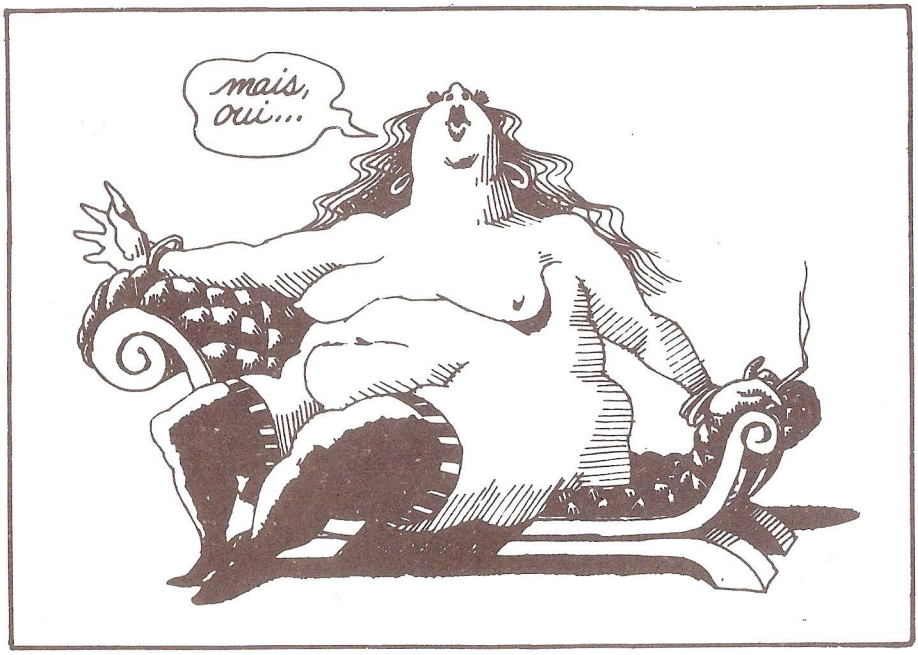
Similarly you can clear a whole line by:

POKE 781,line : SYS 60045

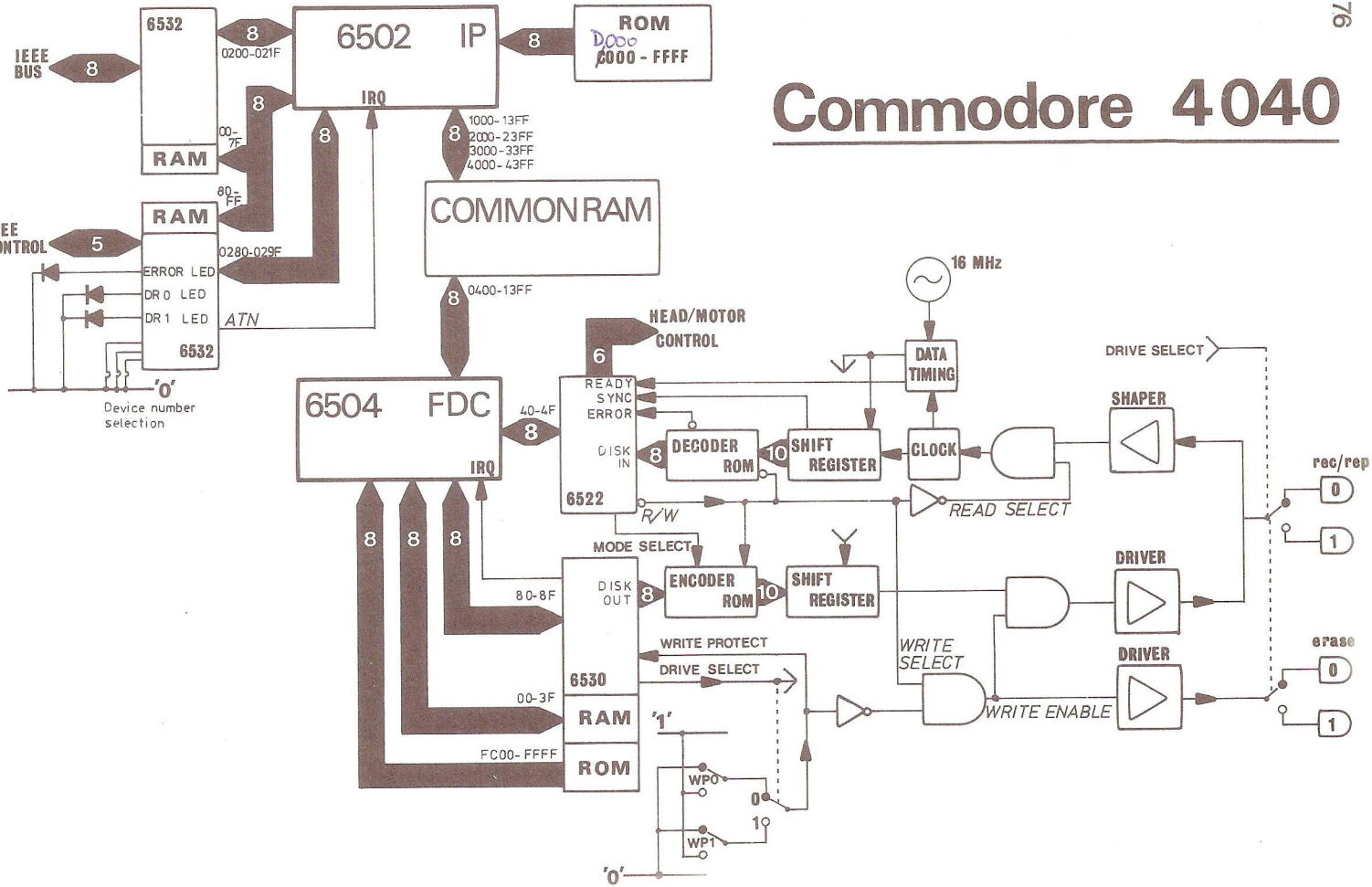
Well that's it for this time. Keep the letters and articles coming. If there is any aspect of the VIC that you would like covered (inside the ROMs, BASIC problems, peripheral problems, etc) let me know. I can not guarantee that I will cover them in the column but I will try.

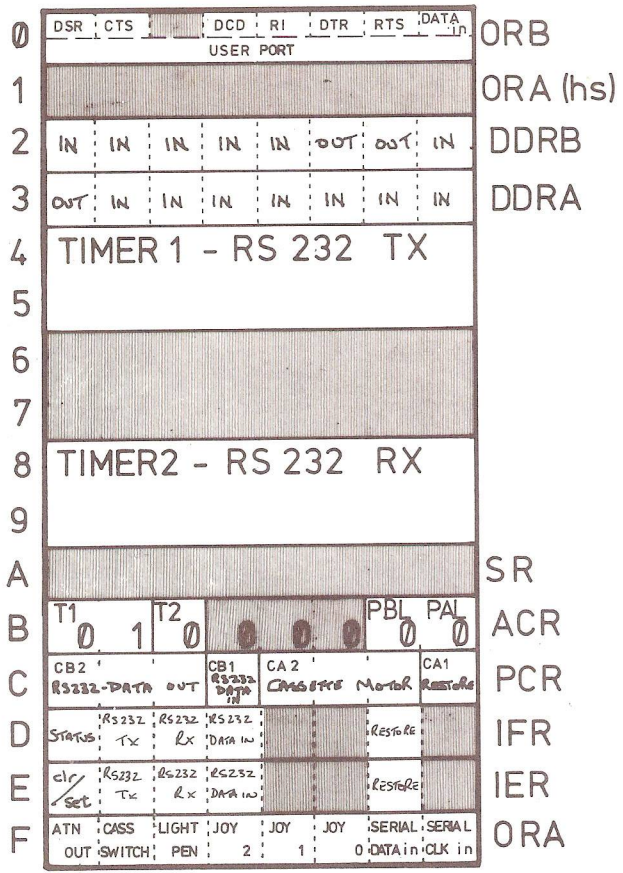
[Late News: B & B Computers has built an add-on for the VIC which increases the column display to 40-columns and increases the memory available from 3K to 35K. The unit contains 32K RAM and a colour writer, as well as a power supply. The price is 220 + VAT. The add-on is available from Beelines, an associate of B & B. - Ed].

--oOo--

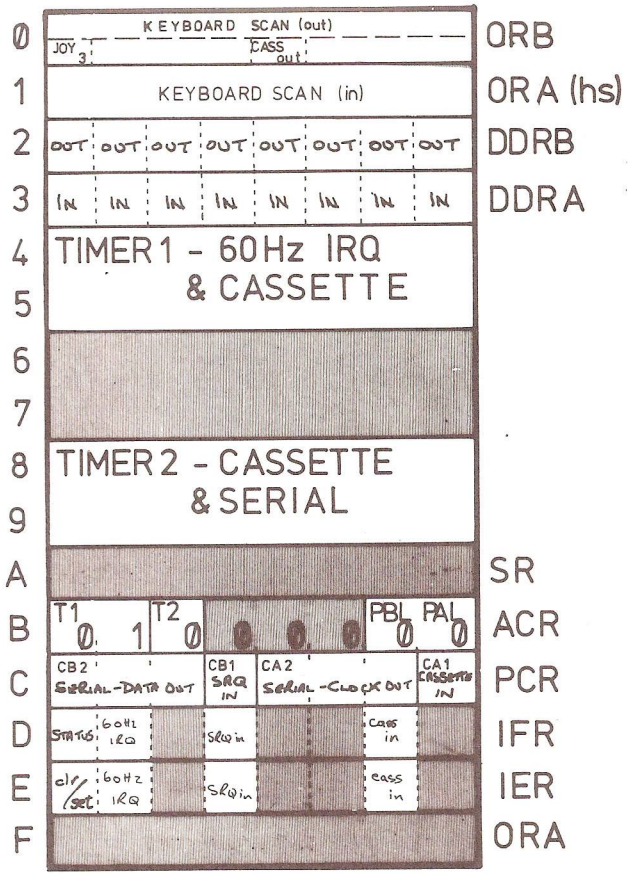


Commodore 4040





VIA #1 - $9110



VIA #2 - $9120

DISK FILE

By Mike Todd

Some of you may be aware that I have been trying to put together a reference manual on the Commodore disk drives. There have been many delays to this publication, and now, with the publication of Raeto West's excellent reference book (reviewed in this Newsletter), the concept of the manual is now somewhat redundant.

Because there is still a need for a technical "underview" of the disk drives, however, I intend to keep a regular column going to cover this side of things. The best place to start is a closer look at how the disk drives are put together using a block diagram of the 4040 disk drive. The 2000 and 3000 series drives are virtually identical and the 8000 series very similar, although the latter does have some significant differences which I'm not in a position to cover here.

You will see from the diagram that the disk unit uses two processors. The 6502 at the top is the INTERFACE PROCESSOR (IP) and handles all IEEE-488 communication as well as handling all character reception/transmission. It also looks after disk organisation such as file handling, block allocation etc. The 6504 is the FLOPPY DISK CONTROLLER (FDC) and simply handles all reading and writing to the disk itself on command from the IP.

The IP uses two 6532 peripheral chips, each of which has 128 bytes of RAM, two input/output ports and a programmable timer. The only interrupt used on the IP is derived from a 6530-2 and interrupts the IP whenever the IEEE ATN line is set, at which point it enters the IEEE-488 handshake routines and procedes to receive data from the bus and then action it.

DOS2 uses a 12K ROM (DOS1 has 8K) and also 4K RAM which is divided into 256 blocks. Some of this is used as workspace, while 14x256 byte buffers (15 in DOS1) are used to pass data to or from the FDC.

The FDC uses only 64 bytes of RAM inside the 6530. This has 64 bytes RAM, 1K ROM, two input/output buffers and a timer. There is also a 6522 containing 2 ports and 2 timers. The FDC uses only one interrupt which comes from the 6530 and which is used for head stepping, timing and the drive motor turn-off delay.

For economy reasons, the electronics is shared between the two drives and the DRIVE SELECT line is used to determine which is the current drive.

To write data to the disk the FDC first checks the WRITE PROTECT line derived from two microswitches WP0 and WP1, which also inhibit the write electronics. The R/W (Read/Write) line is set to WRITE which inihibits the read electronics with READ SELECT and enables the write electronics with WRITE SELECT (gated by WRITE PROTECT). A WRITE ENABLE signal is then generated which turns on the erase portion of the read/write head.

The FDC waits for the correct timing moment (using the READY line) and then puts the data into the DISK OUT port. This first goes through an encoding ROM where it is turned into 10 bits and from there it is presented to a shift register. The resulting 10-bit serial data goes via the DRIVER and then to the disk itself.

Coming back the other way, data from the disk is shaped and presented to the clock circuitry where the timing for the incoming data is extracted. The data is clocked into the shift register and presented to the decoding ROM which presents 8 bits to the DATA IN port on the 6522. If the decoding ROM can't recognise the byte, the ERROR line is set to indicate a "Byte decoding error #24". The FDC uses the READY line to determine when a valid data byte is available on the DATA IN port, at which time it reads this data.

A special SYNC signal is derived from the shift register to identify the start of a block on the disk. This is achieved by writing a 10-bit byte of all "1"s. The encoding process ensures that data bytes are never all "1"s so that this pattern is unique to the so-called SYNC bytes.

The encode/decode ROM is in fact the same ROM but with the R/W line selecting which portion is to be used.

Communication between the two processors is achieved through RAM which is common to both processors. The IP signals to the FDC that it wants a specified block read from or written to by placing a "job control" byte into a job queue. The FDC then does the necessary read/write using the data in the specified buffer. When complete, it returns a code to indicate a successful operation or an error condition.

The block of common RAM is seen by the two processors in different places and the following table shows where these are seen:

IP ADDRESS	FDC ADDRESS	FUNCTION DOS 1.2	FUNCTION DOS 2.1
$1000	$0400	Workspace	Workspace
$1100	$0500	Buffer #0	Buffer #0
$1200	$0600	Buffer #1	Buffer #1
$1300	$0700	Buffer #2	Buffer #2
$2000	$0800	Buffer #3	Buffer #3
$2100	$0900	Buffer #4	Buffer #4
$2200	$0A00	Buffer #5	Buffer #5
$2300	$0B00	Buffer #6	Buffer #6
$3000	$0C00	Buffer #7	Buffer #7
$3100	$0D00	Buffer #8	Buffer #8
$3200	$0E00	Buffer #9	Buffer #9
$3300	$0F00	Buffer #10	Buffer #10
$4000	$1000	Buffer #11	Buffer #11
$4100	$1100	Buffer #12	Buffer #12 (BAM0)
$4200	$1200	Buffer #13 (BAM0)	Buffer #13 (BAM1)
$4300	$1300	Buffer #14 (BAM1)	Workspace

--o0o--

THOUGHT FOR THE MONTH

Thought for the month: there's always one more bug....

--o0o--

STRICTLY FOR BEGINNERS (3)

By Ray Davies

Enough of the theory for the moment, let us get down to some practice. How about a program writing session ?

Try typing this in: (after noting the following abbreviations, which this word processor I am using will not print):

ABB. <clr>= shifted CLR/HOME or CLEAR SCREEN
<cu> = cursor up
<cd> = cursor down
<cl> = cursor left
<cr> = cursor right
<hcsr> = cursor home (top left position on screen)

```
 10 ?"<clr>BEGINNERS PROGRAM
 20 ?:?"--------- -------":REM SHIFTED #
 30 INPUT"WHAT IS YOUR FIRST NAME PLEASE ";N$
 40 ?"<cd>MY NAME IS PET, AND I AM VERY
 50 ?"<cd>PLEASED TO MEET YOU, ";N$
 60 ?"<clr>GOOD BYE FOR NOW !!!":FORI=0TO2000:NEXT
 70 INPUT"<clr>YOUR NAME, PLEASE ";M$
 80 IFM$<>N$THEN?"I DON'T KNOW YOU, ";M$:GOTO100
 90 ?"HELLO AGAIN, ";N$
100 GOTO30
 35 IFN$="ZZZ"THEN?"<cd><cd>GOODBYE, AND THANKS FOR PLAYING
    WITH ME":END
```

This nonsense program demonstrates one or two useful pointers to program writing. First of all it illustrates one way of starting a program off with an underlined title. Then it shows two ways of obtaining a line space on the screen. The second way of doing this is shown in lines 40 and 50. i.e. including a cursor down in the print (or INPUT) statement. This moves the cursor down one line, thereby creating one line space. This can also be done with cursors up, right, left or home.

Also illustrated in line 20 is a REM statement. This is short for REMARK, which means any explanation which you wish to leave in a program, but which the computer will totally ignore. These REMs are very useful in understanding and de-bugging programs (i.e. removing programming faults).

Line 60 illustrates a FOR-NEXT loop with a specific purpose, in this case a time delay, for leaving text on the screen for a particular length of time. You will notice from experience that it takes the PET about one second to count from 1 to 1000. So this particular FOR-NEXT loop leaves the writing on the screen for about 2 seconds, then proceeds to the next line of the program.

Lines 80 and 90 show the effect of an IF-THEN statement. Line 80 checks to see if the new name entered (M$) is the same as the first name entered (N$). If it is NOT the same, then the program prints the message 'I DON'T KNOW YOU' on the screen, then goes to the start of the program. However, if the second name IS the same as the first one, then the program ignores the remainder of line 80 and carries on with line 90 before going back to the start.

The points where the program goes back to the start are simply statements 'GOTO 30'.

You will notice that the final line of the program has not been entered numerical sequence with the rest of the program. This does not matter, as the PET will sort all lines entered into ascending numerical order, and if you now LIST the program, it will indeed be in the right order. Line 35 is in fact a check line to end the program if the first name entered is 'ZZZ'. The program will finish running after a polite farewell message.

As I have not recieved any communications from you yet, I don't know just how advanced you are. But being extremely basic for a moment, let me point out that the word LET is no longer required in front of variables. It used to be necessary to type LET X=6:LET Y=8, but most BASICs nowadays will let you put X=6:Y=8 which is less to type. Also, if you finish a line with the quotes character (") it is not really necessary to type the quotes, so long as you do not want to type anything else on that line. Line 60 above requires quotes, line 40 does not.

As I mentioned in the first of these articles, cassette file handling was my first bugbear, so we will try some of that now.

Next program (don't forget that magic word NEW):

```
100 ?"<clr>STORING DATA ON CASSETTE TAPE<cd>
110 FORI=0TO9:INPUT"SOME NUMBERS ";A(I):INPUT"SOME NAMES";
    N$(I):NEXT I
120 ?"<cd>I AM NOW ABOUT TO STORE THIS DATA
130 ?"ON CASSETTE. PLEASE REWIND THE TAPE NOW.
140 ?"PRESS A KEY WHEN READY
150 GET Q$:IFQ$=""THEN 150
160 ?"<cd>I AM NOW RECORDING THE DATA
170 OPEN 4,1,1,"TEST"
180 FORI=0TO9:PRINT#4,A(I):PRINT#4,N$(I):NEXT I
190 CLOSE4
```

All the names and numbers have now been recorded on cassette tape, under the file name TEST. To recall them back into memory from tape, the following program could be used:

```
200 ?"PRESS A KEY TO CONTINUE
210 GET Q$:IF Q$="" THEN 210
220 ?"<cd>REWIND TAPE NOW, AND PRESS A KEY
230 GET Q$:IF Q$="" THEN 230
240 OPEN4,1,0,"TEST"
250 FOR I=0 TO 9:INPUT#4,A(I):INPUT#4,N$(I):NEXT I
260 CLOSE4
270 FOR I=0 TO 9:?A(I),N$(I):NEXT I
```

It has just struck me that you may not know the purpose of the GET Q$ etc. lines. This is a way of making PET wait for you to press a key. GET causes the machine to get the next key pressed on the keyboard and put the character in the variable (Q$). The second half of the line, after the colon, checks to see if nothing was pressed (Q$ is empty). If so, it loops back on itself indefinitely until something is pressed.

The PET prefers all its files to be labelled. This is why we have a file name (TEST) in this example. The OPEN statement is essential, because it opens a channel from the PET to the specified device, in order that the two machines may talk to each other. Each device has a number - the printer is device 4, the disk drive device 8, and the cassette device 1. This device number is the second number in the OPEN statement. The first number is the file number, which it is up to you to choose. You have a file number because you might actually (usually with disks) want to have several different files open to the same device. In this case our file number is 4, and we use it to tell PET which file to PRINT to and later which file to INPUT from. Note that you cannot use the ? abbreviation for PRINT when handling files. The statement must be entered as PRINT followed by # then the file number then a comma then the actual data to be stored. The third number in the OPEN statement is called a 'secondary address', and its usage depends on which device you are talking to. In the case of a cassette, a secondary address of 1 means you want to open a file to write information into. A secondary address of 0 (line 240) means you want to open a file to read information from. All files should be closed after use, hence line 260. Line 270 merely displays the information which has been read back from the tape on the screen. To prove that the data has in fact been permanently stored on tape, SAVE the program, switch off the PET, switch on the PET, re-load the program and type RUN 200. This runs the program from line 200 onwards, bypassing the lines which store the names and reading them straight from the tape.

I trust that this series is proving to be of some value to you, and I hope to hear from you sometime, with any queries or comments.

--o0o--

DEBUG

Apologies from Nick Higham for an error in his article about the 4022 printer on page 8. CHR$(254) is of course the programmable character and will print out as a space, unless a different character has been defined by printing to secondary address 5 in the usual way.

--o0o--

THE COMMODORE 4022 PRINTER

By Robin Harvey

In the JAN 1982 issue of the IPUG Magazine, Nick Higham made many useful notes about the 4022 Printer and following his theme, I would like to add another.

As Nick mentioned, the Handbook contains a number of errors including the explanation of the programmable character on p29. The correct format is as in the diagram below which, as an example, I have shown with the pound symbol. The bottom row can be used for descenders if required.

Example

128			●	●		
64		●			●	
32		●				
16	●	●	●	●		
8	●	●	●	●		
4		●				
2	●	●	●	●	●	
1						
	26	126	154	154	66	0

TOTALS

A program to demonstrate this is shown below:-

```
 10 REM PRINT POUND SIGN
 20 DATA 26,126,154,154,66,0
 30 OPEN 5,4,5
 40 FOR I=1 TO 6:READ A:A$=A$+CHR$(A):NEXT
 50 PRINT#5,A$
 60 OPEN 4,4
 70 PRINT#4,"ITEM COST ";CHR$(254)" 395.00 PLUS VAT"
 80 CLOSE 5
 90 CLOSE 4
```

There is one feature of the 4022 that I have been unable to make work properly and that is the Secondary Address 3 function i.e. setting the number of lines per page. Has anyone had any success with this feature ?

--o0o--

PULL THE OTHER ONE.....

In case you think I have forgotton to do so, the review of 'The Last One' (see p120 Sept '81) will be completed if ever the distributors make the CBM version available. At the launch I was put through the ritual of signing documents in order to protect their copyright, but I felt at the time it was all part of the show. I trust none of you will part with money for this product until proof of existence and availability is obtained. At least one potential customer has dispaired and written his own code-generator thereby becoming a competitor.

--oOo--

LOOK FOR....

Look out for a low-cost miniaturised Visi-Calc-like package for the 8K PET for £30 on cassette and about £37 on disk. The Cronite Group has launched the package under the name SimpliCalc. It has fewer facilities than its mentor, but can accommodate up to 300 elements on an 8K PET, and up to 3,000 on a 32K model. A maximum of 100 columns can be defined for each worksheet. A version for the VIC-20 is due out in the spring. No other details available at the time of writing.

--oOo--

COMAL UPDATE

By Brian Grainger

In this article I want to add two features on COMAL that I have discovered since my articles in the January Newsletter as well as giving a COMAL program for you to try out.

Firstly let me correct two errors that I found in the previous articles after I sent them to the Editor. I got tied up with my STRING functions on page 13. The equivalent of the BASIC PRINT MID$(NAME$,M,N) in COMAL is PRINT NAME$(M:N). Similarly the equivalent of BASIC PRINT RIGHT$(NAME$,N) in COMAL is PRINT NAME$(END-N+1:N) where END is replaced by the character position of the end character of NAME$. The second error occurred on page 16. In line 0020 of the COMAL program 'RACK' should have read 'RICK'.

With the errors out of the way let us get on with the two new features I have found. In programs one very often wants to add a value to a variable so that in BASIC one says A=A+B. In COMAL with variable names up to 16 characters long this could get a trifle irritating:-

e.g. VARIABLE:=VARIABLE+VALUE

I have found the above line can be written as VARIABLE:+VALUE. Similarly one can subtract a value (or expression) from a variable by using ':-'. It does NOT work for multiplication or division however. You will see some examples of this in the COMAL program following this article.

To finish off this short piece I must mention another bug I have found with COMAL. If one ENTERs a program from tape I found that if one subsequently LISTs the program to tape the PET crashes. The solution is to do a SYS 65511 after the ENTER.

That is it for now except to say that after looking at some utilities from the US COMAL Users Group it would appear that the COMAL OPEN command is more detailed than I have identified so far. I have not investigated fully yet but it IS possible to send disk management commands from COMAL. More about that next time perhaps. In the meantime have fun with MAGIC SQUARES !

--oOo--

MAGIC SQUARES

The game is played with a board of nine cells. Each cell will have either a white dot or a white circle in it. You change the contents of cells from one symbol to the other by pressing one of the keys on the numeric keypad.

With the cells numbered as follows:

```
7  8  9

4  5  6

1  2  3
```

Pressing	1	changes	the	contents	of	cells	1-2-4-5
..	2	..	..	..	..	..	1-2-3
..	3	..	..	..	..	..	2-3-5-6
..	4	..	..	..	..	..	1-4-7
..	5	..	..	..	..	..	2-4-5-6-8
..	6	..	..	..	..	..	3-6-9
..	7	..	..	..	..	..	4-5-7-8
..	8	..	..	..	..	..	7-8-9
..	9	..	..	..	..	..	5-6-8-9

The game is complete when all the cells except 5 are filled with white dots and 5 is filled with a white circle.

You may give up at any time by pressing 0. You will then be shown the complete quickest solution. You may also see the quickest solution after you have solved the puzzle yourself.

For each game the PET will tell you the average number of moves to solve the puzzle shown. For a real challenge try to find the quickest solution which is always two moves less than the par score.

B.D.G.

--o0o--

```
0010 //
0020 // COMAL MAGIC SQUARES
0030 //
0040 // BY BRIAN GRAINGER
0050 //   JANUARY 1982
0060 //
0070 DIM ANSWER$ OF 1, SPACE$ OF 39
0080 DIM SOLUTIONMOVE(9), CELL(9), VECTOR(9)
0090 FINISHED:=FALSE; OLDSTART:=0
0100 SPACE$:="    <39 sp>     "
0110 FOR COUNTER1:=1 TO 9 DO READ VECTOR(COUNTER1)
0120 DATA 229,63,397,219,341,438,355,504,334
0130 WHILE NOT FINISHED DO
0140  EXEC GETSTARTPOSITION
0150  EXEC FILLCELLS
0160  EXEC SOLVEIT
0170  PRINT "<clr>PAR FOR THIS PUZZLE IS";PAR;"MOVES"
0180  EXEC PRINTBOARD
0190  SOLVED:=FALSE; GIVENUP:=FALSE; NUMBEROFMOVES:=0
0200  WHILE NOT SOLVED AND NOT GIVENUP DO
0210   EXEC HITKEYTOGO
0220   EXEC GETMOVE
0230   IF NOT GIVENUP THEN EXEC ALTERCELLS
0240  ENDWHILE
0250  IF SOLVED THEN
0260   PRINT "<clr>YOUR STANDARD OF PLAY IS";
0270   CASE NUMBEROFMOVES OF
0280   WHEN PAR-2
0290    PRINT "PERFECT"
0300   WHEN PAR-1
0310    PRINT "EXCELLENT"
0320   WHEN PAR,PAR+1
0330    PRINT "VERY GOOD"
0340   WHEN PAR+2,PAR+3
0350    PRINT "GOOD"
0360   WHEN PAR+4,PAR+5
0370    PRINT "FAIR"
0380   WHEN PAR+6,PAR+7
0390    PRINT "PATHETIC"
0400   OTHERWISE
0410    PRINT "OF SOMEONE WHO IS GUESSING"
0420   ENDCASE
```

```
0430    REPEAT
0440     PRINT "DO YOU WISH TO SEE FASTEST SOLUTION"
0450     PRINT "(<rvs>Y<off> FOR YES <rvs>N<off> FOR NO)"
0460     INPUT ANSWER$
0470    UNTIL ANSWER$="Y" OR ANSWER$="N"
0480    IF ANSWER$="Y" THEN EXEC SHOWSOLUTION
0490   ELSE
0500    PRINT "<clr>YOU CHICKENED OUT. NOW SEE THE SOLUTION."
0510    EXEC SHOWSOLUTION
0520   ENDIF
0530   REPEAT
0540    PRINT "<clr>DO YOU WISH TO PLAY AGAIN"
0550    PRINT "(<rvs>Y<off> FOR YES <rvs>N<off> FOR NO)"
0560    INPUT ANSWER$
0570   UNTIL ANSWER$="Y" OR ANSWER$="N"
0580   IF ANSWER$="N" THEN FINISHED:=TRUE
0590 ENDWHILE
0600 //
0610 //
0620 PROC GETSTARTPOSITION
0630  REPEAT
0640   NEWSTART:=RND(0,511)
0650  UNTIL NEWSTART<>OLDSTART AND NEWSTART<>495
0660  OLDSTART:=NEWSTART
0670 ENDPROC GETSTARTPOSITION
0680 //
0690 //
0700 PROC FILLCELLS
0710  DECIMALNO:=NEWSTART
0720  FOR BIT:=1 TO 9 DO
0730   BINARYTODEC:=DECIMALNO MOD 2^BIT
0740   CELL(BIT):=SGN(BINARYTODEC); DECIMALNO:-BINARYTODEC
0750  NEXT BIT
0760 ENDPROC FILLCELLS
0770 //
0780 //
0790 PROC SOLVEIT
0800  NUMBEROFMOVES:=0
0810  FOR MOVE:=1 TO 9 DO
0820   IF CHECKMOVE(MOVE) THEN NUMBEROFMOVES:+1;
       SOLUTIONMOVE(NUMBEROFMOVES):=MOVE
0830  NEXT MOVE
```

```
0840  PAR:=NUMBEROFMOVES+2
0850 ENDPROC SOLVEIT
0860 //
0870 //
0880 PROC CHECKMOVE(MOVE)
0890  DECIMALNO:=VECTOR(MOVE); MOVECOUNT:=0
0900  FOR BIT:=1 TO 9 DO
0910   BINARYTODEC:=DECIMALNO MOD 2↑BIT
0920   DECIMALNO:-BINARYTODEC
0930   IF BINARYTODEC<>0 THEN
0940    IF BIT=5 THEN
0950     IF CELL(BIT)=1 THEN MOVECOUNT:+1
0960    ELSE
0970     IF CELL(BIT)=0 THEN MOVECOUNT:+1
0980    ENDIF
0990   ENDIF
1000  NEXT BIT
1010  CHECKMOVE:=MOVECOUNT MOD 2
1020 ENDPROC CHECKMOVE
1030 // [in lines 1070, 1110, 1130 & 1160 use shifted
1040 //  equivalents to those characters in quotes - Ed]
1050 PROC PRINTBOARD
1060  PRINT "<hcsr><7dn>"
1070  PRINT TAB(13),"@@@@@2@@@@2@@@@."
1080  FOR COUNTER1:=1 TO 14 DO
1090   CASE COUNTER1 OF
1100   WHEN 5,10
1110    PRINT TAB(13),"+@@@@[@@@@[@@@@3"
1120   OTHERWISE
1130    PRINT TAB(13),"]    ]    ]    ]"
1140   ENDCASE
1150  NEXT COUNTER1
1160  PRINT TAB(13),"-@@@@1@@@@1@@@@="
1170  EXEC PRINTCELLS
1180 ENDPROC PRINTBOARD
1190 //
1200 //
1210 PROC PRINTCELLS
1220  FOR COUNTER1:=1 TO 9 DO
1230   IF CELL(COUNTER1)=1 THEN
1240    CHARACTERPOKE:=81
1250   ELSE
```

```
1260    CHARACTERPOKE:=87
1270   ENDIF
1280   SCREENPOS:=33582-((COUNTER1-1) DIV 3)*200+
       ((COUNTER1-1) MOD 3)*5
1290   POKE SCREENPOS,CHARACTERPOKE
1300   POKE SCREENPOS+1,CHARACTERPOKE
1310   POKE SCREENPOS+40,CHARACTERPOKE
1320   POKE SCREENPOS+41,CHARACTERPOKE
1330  NEXT COUNTER1
1340 ENDPROC PRINTCELLS
1350 //
1360 //
1370 PROC GETMOVE
1380  REPEAT
1390   PRINT "<hcsr><dn>"
1400   PRINT SPACE$
1410   PRINT SPACE$
1420   PRINT SPACE$
1430   PRINT "<hcsr><dn>"
1440   PRINT "WHAT IS YOUR MOVE"
1450   PRINT "TYPE 1,2,3,4,5,6,7,8,9 OR 0 TO GIVE UP"
1460   INPUT ANSWER$
1470   MOVE:=ORD(ANSWER$)-48
1480  UNTIL MOVE>=0 AND MOVE<=9
1490  IF MOVE=0 THEN
1500   GIVENUP:=TRUE
1510  ELSE
1520   NUMBEROFMOVES:+1
1530  ENDIF
1540 ENDPROC GETMOVE
1550 //
1560 //
1570 PROC ALTERCELLS
1580  CASE MOVE OF
1590  WHEN 1
1600   CELL(1):=1-CELL(1); CELL(2):=1-CELL(2)
1610   CELL(4):=1-CELL(4); CELL(5):=1-CELL(5)
1620  WHEN 2
1630   CELL(1):=1-CELL(1); CELL(2):=1-CELL(2)
1640   CELL(3):=1-CELL(3)
1650  WHEN 3
1660   CELL(2):=1-CELL(2); CELL(3):=1-CELL(3)
1670   CELL(5):=1-CELL(5); CELL(6):=1-CELL(6)
```

```
1680  WHEN 4
1690   CELL(1):=1-CELL(1); CELL(4):=1-CELL(4)
1700   CELL(7):=1-CELL(7)
1710  WHEN 5
1720   CELL(2):=1-CELL(2); CELL(4):=1-CELL(4)
1730   CELL(5):=1-CELL(5); CELL(6):=1-CELL(6)
1740   CELL(8):=1-CELL(8)
1750  WHEN 6
1760   CELL(3):=1-CELL(3); CELL(6):=1-CELL(6)
1770   CELL(9):=1-CELL(9)
1780  WHEN 7
1790   CELL(4):=1-CELL(4); CELL(5):=1-CELL(5)
1800   CELL(7):=1-CELL(7); CELL(8):=1-CELL(8)
1810  WHEN 8
1820   CELL(7):=1-CELL(7); CELL(8):=1-CELL(8)
1830   CELL(9):=1-CELL(9)
1840  WHEN 9
1850   CELL(5):=1-CELL(5); CELL(6):=1-CELL(6)
1860   CELL(8):=1-CELL(8); CELL(9):=1-CELL(9)
1870  ENDCASE
1880  SOLVED:=TRUE
1890  FOR COUNTER1:=1 TO 9 DO
1900   IF COUNTER1=5 THEN
1910    IF CELL(COUNTER1)=1 THEN SOLVED:=FALSE
1920   ELSE
1930    IF CELL(COUNTER1)=0 THEN SOLVED:=FALSE
1940   ENDIF
1950  NEXT COUNTER1
1960  EXEC PRINTCELLS
1970  PRINT "<hcsr><dn>"
1980  PRINT SPACE$
1990  PRINT SPACE$
2000  PRINT SPACE$
2010  PRINT "<hcsr><dn>
2020  PRINT "LAST MOVE WAS";MOVE
2030  PRINT "NUMBER OF MOVES TO DATE IS";NUMBEROFMOVES
2040 ENDPROC ALTERCELLS
2050 //
2060 //
2070 PROC HITKEYTOGO
2080  REPEAT
2090   PRINT "<hcsr><3dn>
2100   PRINT SPACE$
```

```
2110    PRINT <hcsr><3dn>
2120    INPUT "PRESS 'RETURN' TO CONTINUE ": ANSWER$
2130   UNTIL ANSWER$=""
2140 ENDPROC HITKEYTOGO
2150 //
2160 //
2170 PROC SHOWSOLUTION
2180   EXEC FILLCELLS
2190   EXEC PRINTBOARD
2200   FOR NUMBEROFMOVES:=1 TO PAR-2 DO
2210    EXEC HITKEYTOGO
2220    MOVE:=SOLUTIONMOVE(NUMBEROFMOVES)
2230    EXEC ALTERCELLS
2240   NEXT NUMBEROFMOVES
2250   EXEC HITKEYTOGO
2260 ENDPROC SHOWSOLUTION
```

--o0o--

VISICOURSE

If you would like to get the most out of the Visicalc package, there is a course run by the Ibucon Management Centre on March 30th. The fee is £95 + VAT. For further details contact the Conference Registrar, tel: 01-584 2081.

--o0o--

THE COMMODORE ANNUAL

The 'Third International Commodore Computer Show' (formerly and affectionately known as the PET Show) will this year be held on June 3rd-5th at the Cunard Hotel, Hammersmith, London. All communcations regarding the show itself should be made via Clive Booth of Commodore. All queries regarding ICPUG involvement and offers to man our stand, either for a few hours or all three days, should be made to the General Secretary.

--o0o--

CASSETTE MERGE

Nostalgically pondering over the early PET days, it occurred to me that many new-comers, especially disk users may not be aware of one of the earlier revelations. Despite having disks, it is a technique that on special occasions I find useful. Known as the 'Templeton Merge', it came from Brad Templeton of Toronto, Canada by way of Jim Butterfield. The technique merges an ASCII file on cassette into the program in memory by kidding the PET that the incoming file is being entered as if by the keyboard.

To prepare the ASCII file save the program, or part thereof, by:

```
OPEN1,1,1:CMD1:LIST
```

making sure these commands are on a single line. Options include OPEN with a filename and LIST with a range of line numbers. After pressing RECORD & PLAY wait for the tape to stop. Close the file with:

```
PRINT#1:CLOSE1
```

At this point the PET is now back to normal operation.

Now for the merge. When you have in memory the program to which the merge is to occur (if there isn't one, don't forget to type NEW), mount the cassette ready to read. Type POKE14,1:OPEN1, press PLAY as requested and wait for the tape to stop. Now pay attention. Clear the screen and enter three cursor downs followed by:

```
POKE175,1:POKE158,1:POKE623,13:?"<hcsr>"
```

The POKEs are for BASIC2. There is an option whereby one can do OPEN1,1,0,"<filename>". Presumably cassette#2 could also be specified. When the cassette stops, it will end with '?OUT OF DATA ERROR' when it reads the 'READY' line at the end of the file. The merge is then complete.

R.D.G.

--o0o--

COMMODORE STRIKE AGAIN

By Nigel M. Iain.

When Chuck Peddle (daddy of the PET) left Commodore there was much speculation that there was a new machine just round the corner. Everyone now knows that it was the Sirius-1, a truly remarkable personal computer.

Commodore have not been slow to start dropping hints (not to mention press-releases) about their new range of personal computers to include the new VIC-40 (replacing the 40-column PETs) and the Commodore 64. However, much of the information you may read on these machines is based on rumour of forthcoming machines.

When Peddle left Commodore, he took one of Commodore's chief designers (Rolf Pialo) with him. Following the law suits, Rolf went back to Commodore for a salary reputed to be in the order of $400,000. As a result, Commodore have been able to produce another personal computer to add to their range. Already released in the U.S., the Commodore 104 is beginning to take a signifcant market lead and could be set to oust the new Commodore 64 even before it appears.

The 104 uses a newly designed processor (the MOS-technology 0104) which is basically an 8-bit processor with pseudo 16- and 32- bit capabilities. Like many of the bigger mini-computers, the central processor is micro-programmable. This means that, although there is a basic instruction set provided, the micro can have its instruction set altered to suit the application.

All that is left on the 104 is 1024K RAM (addressable directly using extended addressing modes giving total memory capacity of 16Mbytes) which is achieved by having a memory management unit which allows 256 banKs of 64k RAM to be used as if it were a contiguous block of RAM. With the processor running at a staggering 8MHz (and there are rumours of a 16MHz version next year), the internal hardware must put the machine at the top of the range.

Practically all features of the 104 can be changed under software control - the keyboard layout, the screen format, the character set - even (to some extent) the memory layout. The keyboard is a 112-key unit with the function or code for each key stored in a RAM-based decoding table. Therefore any keyboard layout can be set up, and Commodore provide the 104 with no legending on the keyboard so that the user can put his own legends on as required.

Resolution on the screen is 1056x480 points which is used to display anything from 20x10 characters to 132x60; the 104 is initially set up to 80x25. The screen always works in high resolution mode with characters being fetched from the character tables and "drawn" on the screen. This allows high resolution and any size text to be mixed. Screen RAM is not part of the normal memory map and is handled directly by a second processor (a 6502A) so that there is no need to worry about screen RAM conflicts, although memory mapping is available if required. All this results in a very fast screen display, despite the extra processing required. The two processors communicate through a job queue and the main control registers are at $FD00 and $FD01. The former is the character register while the second is the main screen control register.

Interfacing for cassettes is not provided, but with four new interface chips (with 2x16-bit ports, 2x16-bit DACs and 2x16-bit ADCs) it should be possible to configure cassette input/output. However, an IEEE-488 port is provided (fully implemented at last!) while there are two RS422 serial ports. An additional two user ports can be configured up to 16-bits each. Layout is on two main boards and is very neat with only a 2K bootstrap ROM and the microprogram ROM on board. One board is almost empty and can take a variety of ROM/RAM with several ROM switch options available. A memory expansion socket is also provided, but not for normal RAM expansion (which would not normally be required) - instead it is used with the micro programming facility to allow the processor to be set up to emulate almost any micro processor.

Frankly, it is this last facility that is the most exciting since there are 8 emulation packs available which plug into the micro-programming port and turn the 104 into a PET/CBM with BASIC2 or 4, an 8000 or Micro Mainframe, Apple II/III, Superbrain, Atari-800 among others. Future plans include packs for VIC-20 and VIC-40 with ZX81, BBC Micro and IBM-360 packs already under development in the U.K.

On the subject of disk drives, there are no matching drives yet available but the 104 will cope with any of the current range of Commodore drives. It is expected that future versions of the 104 will have integral 4Mbyte drives. These will also be totally "soft" to allow ANY disk to be read.

On the price side, the 104 with all 8 emulation packs is selling for $5790, although a cheaper version (with 64K RAM, a smaller screen and only one emulation pack) should be available very soon at around $1600. The emulation packs cost $420 each and so account for a large proportion of the cost. Commodore have already announced that production of all existing machines is being wound up and that the Micro Mainframe will not be available after 1st June (two months after the 104 is expected to be available in the U.K.).

Lastly, with the development of a range of colour machines in the U.S., there ought to be a full colour 104 available towards the end of the year.

--oOo--

ACC SYMPOSIUM

The ACC will be holding a symposium designed to introduce the hobbyist to the basic principles of data communications and to demonstrate some of the consequences of the technology. Details from P. Whittle, 49, Bartlemas Road, Oxford. Tel: (0865) 721180.

--oOo--

STRING RECOVERY

By Fred Offler

Anyone who has encountered the ?STRING TOO LONG ERROR when recovering data from cassette will know how frustrating it can be and, I have no doubt, many of us have had to resort to re-entering the whole databank. Let us suppose that our program reads:

```
200FOR I = 1 TO 2000
210 INPUT#3, B$(I)
220 NEXT I
```

Let us further suppose that the last line in our program is numbered 2500. A 'string too long error' will result if two data statements have previously been concatenated (you know the drill, let B$(X)=B$(X)+"ABC" so that the resultant string is more than 255 characters long. The trouble usually arises, not when the data is being recorded, but when an attempt is made to recover it !

When the computer grinds to a halt with the '?STRING TOO LONG ERROR IN LINE 210' message, enter: ?I - Let us suppose that the computer responds to the effect that I = 23. Enter

```
3000 FOR I =23 TO 2000
3010 GOTO 210
```

Now start from scratch attempting to recover the data and this time when the computer responds with the error in line 210, merely enter GOTO 3000.

The computer will pick up that portion of B$(23) which exceeded the 255 bytes and will record it as B$(23). All you have to do is to cut the real B$(23) down to size and re-enter it either by means built into your program or by pressing the STOP key and entering B$(23)="....." where '.....' is the new data.

--oOo--

COMMODORE COLUMN

Just as things were beginning to look quiet after the flurry of new Commodore products launched at the PET Show, along comes another wave of goodies. At the Consumer Electronics Show in Las Vegas, CBM introduced their video game microcomputer with a music synthesiser. This first entry into video games by Commodore can imitate several musical instruments. US price is $150. Two additions to the VIC family were shown priced at $400 and $600, but don't expect to see them in Europe this year because of TV standard conversion problems.

Finding out about the next product has been like putting a jigsaw together with less than half the pieces. The Model 64, as I shall call it (I don't think there is an official name for it yet) will make its debut at the Hanover Fair starting April 21st. It is basically a 64K system with a 6509 8-bit processor (is it their equivalent of the 6809 ?) and the ability to run packages already developed for other machines such as Atari 800, TRS-80, Apple II, and even the IBM and Sirius machines. The 6509 processor has memory management built into the chip and can in consequence address up to 256K of RAM. The 'copycat' feature is possible by a card carrier at the rear which enables the microprocessor to be replaced by, for example an 8088 to emulate the IBM machine, although it is not a true emulation. Price is a mere $595. Reports are confusing with these new products as one is never sure if the same model is being discussed from a different source, but 40- and 80-column models are mentioned with colour and advanced graphics. But are these the two additions to the VIC family. I ask myself. It is early days yet, but expect to see the demise of the 4000-series.

On disk matters, the 2031 single disk drive is £ 395 + VAT while the single floppy drive for the VIC is £ 396. VAT inclusive. Up market, two Winchester drives are to be introduced at COMDEX'81. The first is a 5+1/4" Winchester drive rated at 6.4M-byte followed by another at 9.6M-byte both built and packaged to Commodore

specifications and to be priced considerably lower than the competition. Commodore are also to manufacture a 5M-byte Winchester scheduled for April. also competitively priced. Finally there is a 43M-byte 1/2" magnetic tape drive. The latter two devices are reported to be no more than 2.2" high.

Arfon's expansion board for the VIC has become the first Commodore Approved product for the VIC. Apparently Arfon were contracted to debug a consignment of 1500 VIC (Seikosha) printers from Japan because of a timing problem in the software arising from different frequency standards in Europe to Japan & the US.

As mentioned in my editorial Commodore's official user group has lost its Editor and magazine to Nick Hampshire Publications. Expect to see further changes...

Finally. there are rumours of a 16-bit machine in the pipeline.

R.D.G.

--oOo--

TRICKS WITH LIST

Last month we mentioned that POKE19.157 followed by LIST did some every odd things. The question was - WHY ?

Generally. whenever the PET prints a string of characters it needs to set up a pointer to the string and calculate its length. It is using this "string descriptor" that the actual string is output.

As part of the software to do this, there is a string descriptor stack starting at location 22 in BASIC2/4 and a stack pointer at 19. It is this stack pointer which is corrupted with the POKE and when the output routine tries to access the last descriptor on this stack it basically gets garbage. This means that the output routine will fetch a string starting almost anywhere and of random length. Hence the garbage you get when LIST is used.

The POKE value is actually the zero page location of the start of the string descriptor and the effect could be predicted by working out what bytes one would expect at that location.

M.R.T.

--oOo--

ICPUG PROJECTS

This announcement is intended to bring members up to date with the situation regarding the machine code course and the hardware projects.

Machine Code Course.

There are two options now open to members:

a) Phased lessons

Lessons will be posted weekly. Required from members:
£ 1.00 (Cheque or Postal Order made payable to F. Offler);
12 Stamped (12+1/2p) addressed envelopes (24 x 16.5cm).

b) Bulk Lessons

The complete course will be posted at the time of registration. Required from members:
£ 1.00 (Cheque or Postal Order made payable to F. Offler);
1 stamped (22p) addressed envelope (not less than 24 x 16.5cm).

Members who require a folder for their papers should add a further £ 1.50 to the payment to cover the cost of folder, envelope and postage.

We intend to have a statistical analysis carried out on the course assessments received so far and this will be published in due course. First indications are that we have achieved a 90% success rating.

DATA SHEETS FOR HARDWARE PROJECTS.

Prototypes of the equipment (p157) have been in use for some time and have proved successful. The weather and other commitments have prevented the project committee from meeting to finalise the arrangements. In any event the 'interface interface' is being modified to include both a real time clock and buffering on all outputs.

Most people who registered for the machine code course have envelopes lodged ready for the release of details of the project scheme. Any other member who is interested in the projects is advised that it is only necessary at this stage to supply an ordinary sized stamped addressed envelope.

All enquiries about the machine code course and/or the projects should be addressed to:

Fred Offler,
37, Brooklyn Road,
South Norwood,
LONDON, SE25 4NH

and please, do include a stamped addressed envelope if you want a reply.

--o0o--

PIRACY RULES - OK ?

In the US a federal court has ruled that object-code programs in ROM cannot be copyrighted, even though the source code was copyrighted. One of the bases for the decision was that the object code is not a true copy of the source code, because it is not in a human-readable form.

--o0o--

THE LONDON COMPUTER FAIR

This function will be held on the 15th, 16th & 17th April 1982 at the Polytechnic of North London, Holloway Road, London, N7. ICPUG have a stand on all three days of the fair and we need volunteers to help us on the stand.

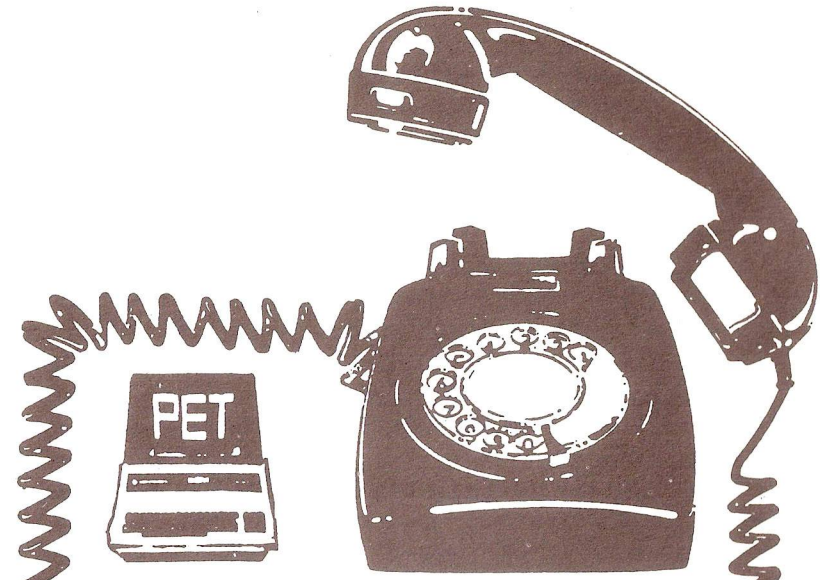
On the Saturday we are running a VIC games and graphics ARCADE and need several people to act as stewards.

If you help us, we will meet your travelling expenses, but let us know soon, as it is strictly first come, first served. Contact Stephen Rabagliati by telephone:

Home 0442-44743
Work Radlett (09276) 7141 extn 39

If you cannot help, do please come and see us at the Fair.

--o0o--



CLEARSONS LTD.

Cash & Carry Computer
and
Word Processing Supplies

All types of computer stationery

Listing paper Ex-stock

Free quotation for your letterheads, invoices, statements etc.

Appointed dealer for

COMMODORE
micro computers

JUST LIFT THE PHONE

Farnborough, Hants

518022 & 518717

30 Camp Road, Farnborough, Hants.

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