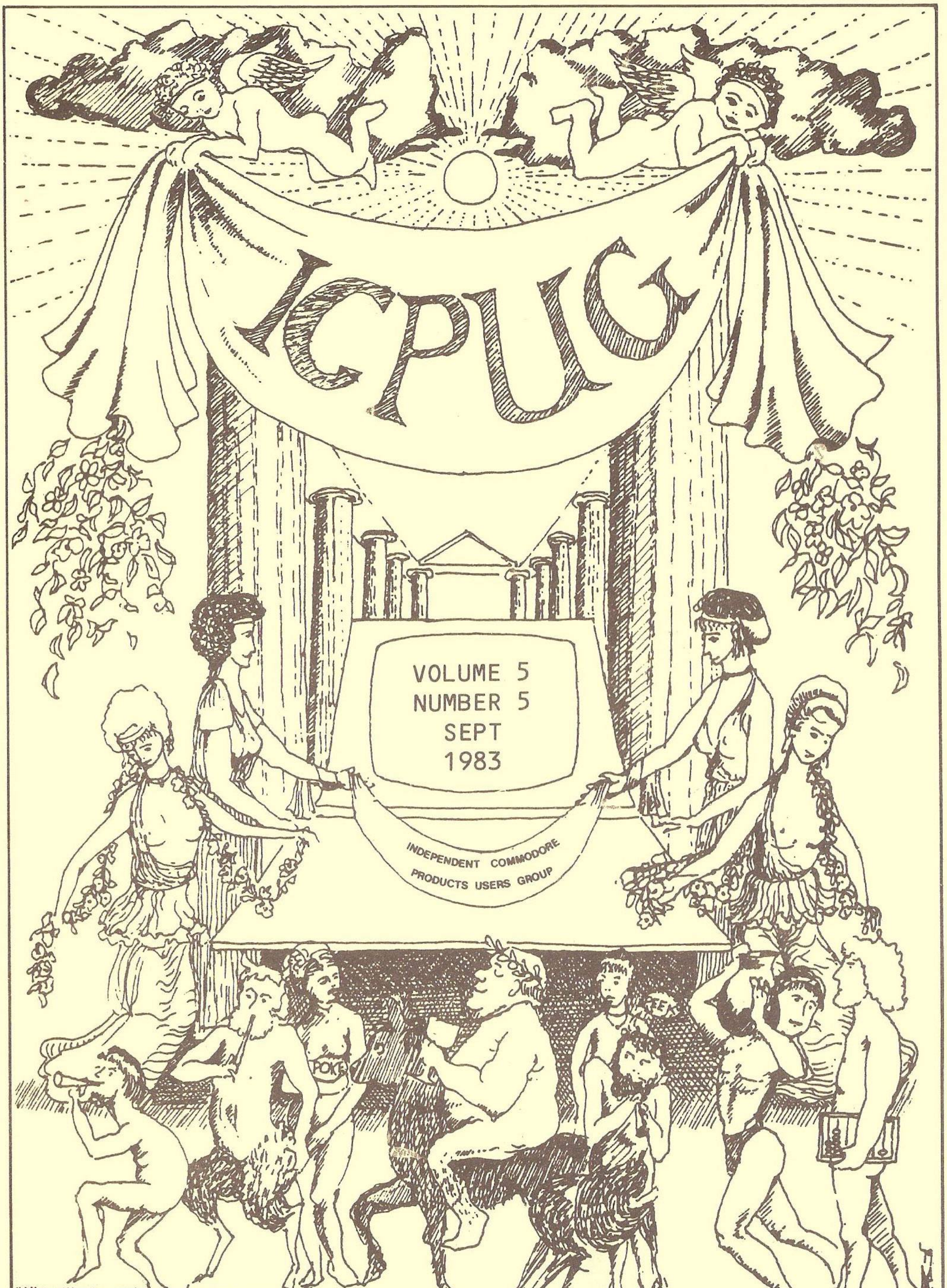

ICPUG
VOLUME 5
NUMBER 5
SEPT
1983
INDEPENDENT COMMODORE PRODUCTS USERS GROUP
POKE
PEEK

HONORARY NATIONAL OFFICIALS

CHAIRMAN	Wing Cdr. Mick Ryan 164 Chesterfield Drive Riverhead Sevenoaks, Kent TN13 2EH Telephone: Sevenoaks (0732) 453530
REGIONAL CO-ORDINATOR	Terry Devereux 32 Windmill Lane Southall Middlesex UB2 4ND
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
VIC CO-ORDINATOR	Mike Todd 27 Nursery Gardens Lodgefield Welwyn Garden City Herts AL7 1SF
DISCOUNTS OFFICER	John Bickerstaff 45 Brookscroft, Linton Glade Croydon CRO 9NA Telephone: 01 651 5436
ASSISTANT EDITOR	Tom Cranstoun

INDEPENDENT COMMODORE PRODUCTS USERS GROUP

VOL 5
No 5

Newsletter

SEPT
1983

Europe's first independent magazine for PET users

EDITOR'S NOTEBOOK

It was with a sigh of relief that I read the letter from Ron Barrett offering his services as Editor, for rarely at the A.G.M. do we have more than one person nominated for any given post. Many jobs go vacant after the Meeting and have to be filled later when the Committee 'send the boys round' looking for a volunteer or two.

Having been editor since ICPUG began some five years ago, when I was the 19th person to join, I have seen the Newsletter grow from a few stencilled hand-typed pages to its present day size. This has only been possible by the contributions made by you, the membership, and it would be unfair to single out individuals for special mention, but then their names appear regularly in these pages, so you know who I mean. Thanks lads, keep those articles coming. On the thank you theme I would also like to record my thanks to the various assistant editors and other helpers that have aided me in the task, not forgetting the final polish provided by the printers.

Do make an effort to attend the AGM, it may not be your idea of entertainment, but it is only once a year and it does provide an opportunity to meet the national committee and put a face to the names.

For the next issue, material for publication should be sent to Ron Barratt at 2, Hazelwood, Windmill Hill, Brixham, Devon.

R.D.G.

--oOo--

MEMBERS PRIVATE SALES AND WANTS

Wordcraft-20 cartridge for Vic including 8K expansion memory also usable in BASIC, for sale at £ 80 or offer to Colin Pearson, 167, Nantwich Road, Audley, Stoke-on-Trent, ST7 8DL. Tel: (0782) 720753.

Vic-20 Gorf cartridge £ 12.50, games tapes: Moons of Jupiter (Romik) £ 3.50, Skyhawk (Quicksilva) £ 3.50, Krazy Kong (Anirog) £ 4.00, Traxx (Llamasoft) £ 3.50, 16K RAM Pak (Stonechip) £ 22.00. Contact D.K.Parkinson, 84, Abbotsbury Road, Broadstone, Poole, Dorset. Tel: Broadstone 694226.

--oOo--

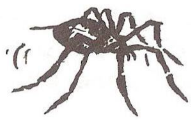
VISICALC - The @MIN Function

By Brian Grainger

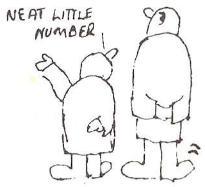
Before getting to the meat of this article I want to relate the experiences of Allan Thompson and VISICALC. Allan has recently added a CP/Maker to his 8032/8050 system. This, together with a 'loader' program, allowed the 96K Visicalc version to operate. This apparently gave 73K of free memory. Allan found, however, that using a fairly large sheet and having 40K free space available rows below 190 could not be accessed. He suggests that the problem may be that Visicalc on an 8096 only allows 69K free and this may be impacting on the CP/Maker. I personally do not find this too convincing as from the numbers quoted it has only used up 33K of memory and I cannot see how the addition of a few rows could cause the memory to go over 69K however large the sheet. If anybody has any thoughts on this problem let me know (73, Minehead Way, Stevenage, Herts).

Allan also provided the trigger for the subject of this article, namely the @MIN function. As all Visicalc users will know, @MIN will provide the minimum of a range of values. What Allan wanted to do was find the minimum of a column of numbers where, because of other constraints, some of the entries are blanks. For numeric functions a blank entry, or a label entry, will provide a value of 0 which could incorrectly cause @MIN to return 0 when some other value is appropriate. I had to contrive a solution to this problem but it works nevertheless. It makes use of the @COUNT function which is the only basic function which varies on whether an entry has a number or a blank in it. @AVERAGE also gives different answers but that makes use of @COUNT anyway. Here is an example sheet printout together with the formulae printout that produced the sheet.

```
         A     B     C     D     E
 ------------------------------------
 1!      5     3BRIAN            7
 2!      5     3     3     3     3
```



```
>E2:(@MIN(D2,E1)-D2)*@COUNT(E1...E1)+D2
>D2:(@MIN(C2,D1)-C2)*@COUNT(D1...D1)+C2
>C2:(@MIN(B2,C1)-B2)*@COUNT(C1...C1)+B2
>B2:(@MIN(A2,B1)-A2)*@COUNT(B1...B1)+A2
>A2:+A1
>E1:7
>C1:"BRIAN
>B1:3
>A1:5
/W1
/GOC
/GRA
/GC6
/X>A1:>A1:
```



Let us see how the idea works. Row 1 is the row for which I want a minimum value calculated. I include a blank entry at C1 and a label entry at D1 just to confuse things. Row 2 is a working row of figures calculated as follows.

1) Set A2=+A1

2) Calculate a 'running' minimum in row 2 against the corresponding elements of row 1

3) B2 is thus the minimum of A2 and B1 provided B1 is not a blank or label. If B1 is a blank or label then B2 will be equal to A2.
B1 will be a blank or label if @COUNT(B1...B1)=0. If @COUNT(B1...B1) is 1 then B1 contains a true number

4) From 3) the expression for B2 can be calculated -
  B2=(@MIN(A2,B1)*@COUNT(B1...B1))+(A2*(1-@COUNT(B1...B1))
This can be simplified to -
  B2=(@MIN(A2,B1)-A2)*@COUNT(B1...B1)+A2

Note that the range in the @COUNT function is essential. Do NOT type @COUNT(B1) as, for some reason known only to the software writers, the result will ALWAYS be 1.

5) B2 can be replicated from C2 to E2 with all parameters as 'RELATIVE'.

6) The final result will be given under the last entry (E2).

I leave the reader to generalise the above example to a column of figures. If you always keep a working row below the row to be minimised and a working column to the right of a column to be minimised things will work OK (with normal recalculation order). The working rows and columns can always be set outside any area which has to be printed to paper so the technique is useful for real applications.

--oOo--

Then I left the paperless office
to join the cashless society....

NEW BOOKS ON THE WAY

Two new books in preparation are 'Programming Vic-20' and 'Programming Sound, Graphics, (and More!) on Your Commodore 64'. The former is by Raeto West, who will be known to many of you, and the latter is by his brother Marcus. Both books are large-format spiral bound and I hope to review these titles when they are available. UK distribution to the trade is by Edward Arnold (Publishers) Ltd., Woodlands Park Avenue, Woodlands Park, Maidenhead, Berks, SL3 3LX.

R.D.G.

--oOo--

SOME MORE MEMBERS QUERIES

By Mike Todd

CQ CQ CQ

I know that there are many amateur radio enthusiasts in ICPUG and I often get queries relating to the use of the PET, VIC and 64 in the hobby.

There are several commercial packages for RTTY and CW available, but they are rather expensive. There are also a couple of rather simple RTTY programs around, but does anyone have a reasonably sophisticated package for the VIC and/or 64 which could be passed on to interested members ?

Ian Greenshields (G4FSU) would also like to know if anyone has attempted to receive the UOSAT (University of Surrey's) satellite signal on the VIC.

IMPROVING VIC COLOURS

In the early days of the VIC, rumours abounded about a simple hardware modification to the VIC which would improve its colour significantly.

This was reported to have been devloped by Ira Curtis Coleman and I wonder if anyone has the details as there are several members interested.

SOUND ON THE 8032

The CB2 line on the 8032 is fed to the small, internal "loudspeaker" and is used for the end of line "beep" and the CHR$(7) bell.

It should therefore be easy to use the conventional sound generation technique with the 8032:

```
POKE 59467,16: POKE 59466,15: POKE 59464,pitch
```

Unfortunately, I have had reports that this doesn't always work reliably and, not having an 8032, I've been unable to check this out. Has anyone had similar problems - if so, how did you solve them? I know that the internal signal is gated by the diagnostic sense line, and so could be turned off by this line - is it possible that the problem could be tied up with this ?

EPROMS & THE 4040

The 4040 disk drive has three ROM sockets on board and it is very tempting to make minor modifications to the DOS to correct bugs, or simply to experiment.

Unfortunately, because of the way that the 4040 shares its RAM between the interface processor (6502) and the floppy disk controller processor (6504), the chip select lines (CS2 on pin 21) on the 2332 ROMs are actually used.

On the 2532 EPROMs, this pin is the Vpp (programming pulse) signal and so 2532 EPROMs cannot be used. So, does anyone have any ideas how this can be overcome easily - would 2732s be the answer, or would a hardware mod work ?

EPROMS IN GENERAL

The subject of ROM and EPROM compatibility is quite a complex one. For instance, there are a couple of different (and totally incompatible) versions of EPROMs with the same number !

If anyone who has some experience of the problems of interchanging ROMs with EPROMs would like to write an article (or articles) for the Newsletter, I'm sure that there would be a wide interest.

Please address all replies to me and I'll pass them on.

--oOo--

REVIEW - PLAYBYTE

If you write your own programs the day will almost certainly come when, in a fit of enthusiasm, you type RUN instead of SAVEing it first. Owing to a bug/typing error, the keyboard goes dead and even with the RESTORE key of the Vic-20 or C-64 you have just lost 3 hours of typing. In dispair you switch off and back on again and start again.

What you needed of course was a reset button like on the CBM700. The answer comes from 'Playbyte' which is a push-button on a small circuit board which plugs into a Vic-20 or C-64 and gives one access to the reset line. A brief press of the button and 'bingo', your keyboard is now alive again and entering LIST reveals... nothing. Reset invokes the NEW/CLR sequence. Fortunately, 'Playbyte' comes with a short program on cassette, which when loaded and called restores the program pointers.

As can be seen, 'Playbyte' will recover a program if you type NEW and then wish you hadn't. This is equivalent to the UNNEW or OLD command on some computers. The program can also be useful if you have disabled the STOP key and now wish to break out of your program. Users will no doubt think of other uses for this.

For the curious, you may be wondering why, when 'Playbyte' was loaded, it did not over-write your program. 'Playbyte' is written to reside in the bottom of the stack area, a most unlikely place for your BASIC program to be !

I was intrigued by the track layout on the pcb, considering it only contained a push-button switch, perhaps there is scope here for further facilities.

'Playbyte' comes with a typed instruction sheet in two versions, one for Vic-20 and another for the C-64. For the price of £ 10.50 'Playbyte' is certainly worth having and for ICPUG members, the price is only £ 7.50.

Orders should be sent to 'Playbyte', 5, The Spinney, Fleet, Hants, GU13 8EP.

R.D.G.

--oOo--

```
*****************************************

10  STILL WAITING TO AFFORD YOUR COMMODORE 700,
    500 OR EVEN C-64?

20  IF A VISIT TO YOUR BANK MANAGER IS
    LIKE A RETURN WITHOUT GOSUB ERROR
    THEN STOP.

30  BREAK OUT OF THE CONTINUOUS LOOP.
    GET AN INCOME FROM YOUR PROGRAMMING
    SKILLS.

*****************************************
```

If you think that your programming ability could provide you with an income, but are hampered by lack of expensive hardware, then **Mr.Software** may have the answer.

We're currently looking for programmers, skilled in both "BASIC" & 6502 Assembly Language who have a flair for graphics and a lively imagination. If the description applies to you, why not send **Mr.Software** a sample of your program at the address below.

If your work shows talent and promise we will lend you the hardware to enable you to earn an income from your programming skills.

Mr.Software
18-20 Steele Road,
Park Royal Industrial Estate,
London NW10.

TEACH YOURSELF PROGRAMMING - LESSON 1

By Brian Grainger

NOTE:- This course is designed to teach beginners to program in COMAL or BASIC. Where there are differences between the two languages the COMAL points will be given in the main text. A note reference (e.g. *5) will be given in brackets and users in BASIC should refer to the relevant note at the end of the article.

GETTING STARTED.

When you switch your computer on you can do one of two things. Tell it to perform some operation immediately or give it a list of operations to follow when you tell it to start. For example you could tell it to add 5 and 6 and print the result by:-

PRINT 5+6 and pressing 'RETURN'

This is called working in direct mode. Alternatively we could tell the computer to do the following instructions when we tell it to start (i) get two numbers, (ii) add them up and (iii) print the result. We could do this by (press 'RETURN' after each line):-

```
   10 INPUT A,B
   20 C=A+B
   30 PRINT C
```

This is called working in program mode and you will notice the computer does not do anything except store the instructions. We can tell the computer to obey the instructions by typing RUN and pressing 'RETURN'.

When an instruction is given WITHOUT a line number the computer will try to obey it immediately. When an instruction is preceded by a line number the computer will store it and not execute it until told to do so. A computer program is a set of instructions stored in the computer which when RUN will be executed in line number order (unless the instructions tell the computer otherwise).

When we write a program we usually want to make sure there are no programs already stored in the computer so to do this we type NEW followed by pressing 'RETURN'.

From now on I will not mention the pressing of 'RETURN'. Whenever you have finished typing a line (and corrected any typing errors) you will press 'RETURN' to tell the computer to accept the line.

When we have a new storage area we can write our instructions giving each one a separate line number. It is usual to leave gaps between the numbers so that we can add some lines in between, should we need to, to correct mistakes. The computer will automatically store your lines in numerical order no matter what order they were typed in.

With COMAL (but NOT BASIC) we can get the computer to prompt line numbers automatically by for example:-

```
    AUTO 100,10
```

This will prompt a line number 100 and then after a line has been input will prompt 110 (=100+10) and then 120 (=110+10) and so on. Pressing 'RETURN' next to a line number will turn AUTO off. Of course you can use numbers other than 100 and 10. If you type AUTO by itself it will assume you have typed AUTO 10,10.

When we have input our program we will want to store it on a cassette or disk so that we do not lose it when we turn the computer off. To save a program we say in direct mode:-

```
   SAVE "0:PROGNAME" if we are saving to disk or
   SAVE "PROGNAME",1 if we are saving to cassette
   (*1)
```

When we switch the computer on again and wish to have the program stored in memory again we would reload it by:-

```
   LOAD "0:PROGNAME" if it is on disk or
   LOAD "PROGNAME",1 if it is on cassette
   (*2)
```

When we have a program in memory we may wish to look at it. We can do this by LIST in direct mode. This will list the program in line number order until the end or until the 'STOP' key is pressed. If we only wanted to list a portion of the program we would say for example:-

```
   LIST 40-590
```

This would list all lines with numbers beetween 40 and 590. With COMAL, lines when listed will be indented. We will come to this in a later article but it does cause a problem if we want to list a few lines and then correct some of them by moving the cursor to the appropriate line and changing it before pressing 'RETURN'. We do not want all the extra spaces so when using COMAL (NOT BASIC) we would use EDIT instead of LIST when we want to edit lines. You can say which lines are displayed with EDIT just as you can with LIST.

Sometimes we find that we want to remove a lot of lines from a program. In BASIC we would have to type each line number and press 'RETURN' after it to remove the line. In COMAL we can also remove a block of lines by say:-

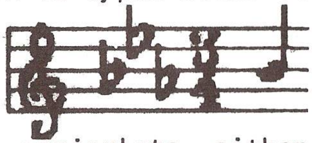
```
   DEL 20-560
```

This will remove all lines between 20 and 560 from the program.

Another thing we sometimes want to do is change all the program line numbers. We can do this in COMAL (but NOT BASIC) by say:-

```
   RENUM 200,10
```

This will renumber all the program lines so that they start at 200 and go 200, 210, 220, ... You can use any numbers instead of 200 and 10 of course. If RENUM is typed alone it will assume you have typed RENUM 10,10.



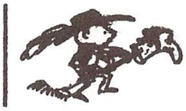
NUMBERS AND CHARACTERS.

Programs are generally used to manipulate either numbers or characters. For example a calculator program would manipulate numbers, a word processor would manipulate characters. You can of course type numbers or characters in a program just as you do on paper but more usually we want to store numbers and text so we can manipulate them. A variable is a stored element whose contents we can change. We have numeric variables or text variables. An identifier will be used in the program to refer to the variable. We can view a variable as a storage box into which you can store different numbers (or characters). The identifier is the label on the box by which we refer to it. I shall now outline the rules for naming variables.

Real Variables.

These are used to store any type of number whether it is a whole number like 1 or 10, or a decimal number like 5.6 or numbers a number like 10 may be stored as 10.000000001. This can sometimes cause errors in calculations or the way a program flows if you are not careful. The advantage is that any number between -1x10↑(-38) and 10↑37 can be stored to about 9-digit accuracy.

A real variable identifier can be up to 16 characters long, the first of which must be alphabetic but the others can be alphabetic, numeric or ',\,[,],←
e.g. NUMBER, AREA, CIRCLE'RADIUS.
(*3)



Integer Variables.

Integer variables are used to store whole numbers such as 1, 10, -36. The advantages are that they take up less space and are stored exactly. The disadvantage is that we can only store numbers in the range -32768 to 32767. An integer variable identifier is named like a real variable but has a # following the characters, e.g. INTEGER#, MILES#
(*4)

String Variables.

String variables are used to store text. In fact they can store ANY collection of characters, not necessarily alphabetic. A string variable name follows the same rules as real variables but have a '$' appended.
e.g. STRING$, NAME$
One other complication with COMAL (NOT BASIC) is that we have to say how many characters each string variable can hold. The value given will be a maximum number. We do this with a dimension statement.

e.g. If we know NAME$ will store a maximum of 10 characters we would say:-

```
DIM NAME$ OF 10
```

We can dimension more than one string variable on one line as follows:-

```
   DIM NAME$ OF 10, STRING$ OF 15
```

The advantage of dimensioning strings like this is that it is much easier to refer to parts of text stored in a variable and VERY much faster to process. Note that a string variable MUST be dimensioned before it is used and it cannot be redimensioned so you must consider what the maximum number of characters will be very carefully. BASIC does not require a dimension statement but strings are limited to 255 characters long.

Boolean Variables.

These are in fact no different from real variables but the contents can be regarded as specifically TRUE or FALSE. In fact FALSE has a value of 0 and any non zero number is regarded as TRUE although when a variable is set to TRUE it is given the value of 1 (-1 in BASIC).

ARRAYS.

Sometimes it is very convenient to group different variables under one name. For example you may hold a record of your bank account and each cheque amount would hold similar information. If we called the various amounts CHEQUE'AMOUNT1, CHEQUE'AMOUNT2, etc. it would become extremly tedious to process each amount. For example to print each to screen for display purposes would require one print statement per amount. We can get round this problem by calling the different cheque amounts CHEQUE'AMOUNT(1), CHEQUE'AMOUNT(2) etc. This time we can print all the amounts with one statement say:-

```
   FOR I=1 TO 15 DO PRINT CHEQUE'AMOUNT(I)
```

A variable such as CHEQUE'AMOUNT above is called an Array variable and the number in brackets is known as the index value to a particular element in the array. It is also possible to have variables such as VALUE(2,3). This time we can regard VALUE being made up of a rectangle of boxes, each with a different number stored in it. VALUE(2,3) is used to index the box in the 2nd row and 3rd column. We could continue like this adding more than just rows and columns but in practice two indices are usually enough.

Before we can refer to array variable elements we must tell the computer to reserve space for all the elements. We do this by the DIM statement again. We must do this whether we are using BASIC or COMAL. An example of a DIM statement is:-

```
   DIM VALUE(1:10,5:8)
   (*5)
```

This reserves space for 40 elements of the array VALUE. There are 10 rows numbered 1 to 10 and 4 columns numbered 5 to 8. Any attempt to refer to an element outside these rows or columns will cause an error. The 'row' or 'column' is called a dimension of the array. For each dimension a lower limit and upper limit has to be defined. These are the two numbers separated by a ':'. Each dimension range is separated by a ',' and the whole is enclosed in brackets after the array name. Array names follow the same rules as normal variables. We can have real arrays, integer arrays and string arrays. In the latter case each element of the array will have the same maximum number of allowed characters which is dimensioned at the same time as the array itself.

e.g. DIM NAME'ARRAY$(1:5,1:6) OF 10

In COMAL the lower and upper dimension limits must be whole numbers but they can be negative if required. If the lower limit is 1, the '1:' may be omitted from the dimension statement.

e.g. DIM VALUE(5,6) is the same as DIM VALUE(1:5,1:6)

EXPRESSIONS.

GO! GO! GO!

Now that we have defined a variable what can we do with them? Basically we can set them to a constant value such as:- VALUE=5 or
NAME$="BRIAN" (note the use of '"' to enclose text information)
FINISHED=TRUE
Note that in COMAL such statements will be listed as NAME:="BRIAN" etc. with an added ':'. This is to

distinguish the assign command(:=) from the equals operation(=) which we will talk of shortly. BASIC makes no such distinction.

We can combine variables with other variables or constants into expressions.

Numeric expressions.

We can combine constants or variables by adding them(+), subtracting them(-), multiplying them(*), dividing them(/), or exponentiation(↑). In COMAL we also have integer division (DIV) and remainder from division (MOD).
e.g. 7 DIV 2 = 3 and 7 MOD 2 = 1.

String expressions.

The only way of combining string constants or variables is concatenation(+), adding one string to the end of another.
e.g. If A$="ABC" and B$="123" then A$+B$= "ABC123".

In order to manipulate strings in COMAL one refers to the characters of the string directly. e.g. If A$="ABCDEFGH" and we wish to set A$ to "ABCD12GH" we would say A$(5:6)="12". Similarly to set B$="ABCD12GH" we could say B$=A$(1:4)+"12"+A$(7:8).

When referring to strings like this we refer to a single character by say A$(5) not A$(5:5), although the latter still works.
(*6)

Boolean Expressions.

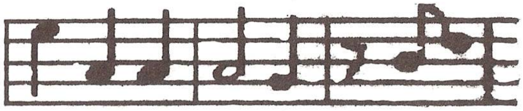
These can be formed by combining Boolean constants or variables using NOT, AND, or OR.

If A is TRUE then NOT A will be FALSE and vice versa. A AND B is TRUE only if both A and B are TRUE. Otherwise it is FALSE.

A OR B is FALSE only if both A and B are FALSE. Otherwise it is TRUE.

Boolean expressions can also be formed from numeric and string constants and variables by using relational operators. These are equal(=), not equal(<>), less than(<), greater than(>), less than or equal(<=), greater than or equal(>=). In additon COMAL has a string relational operator IN. Some examples follow:-
A=2 is TRUE if A=2, otherwise it is FALSE
A<>5 is TRUE if A is not equal to 5. Otherwise it is FALSE.
B$ IN "ABCD123" will be TRUE if B$ consists of some consecutive characters of "ABCD123" such as "ABC", "CD1", "12". If B$ is say AB3 then the expression will be FALSE. (Note the IN function is actually a numeric function - the value it returns is the position in the string of the first character of B$).

NOTES FOR BASIC PROGRAMMERS



*1) Saving to cassette is done by SAVE"PROGNAME". Saving to disk by SAVE"0:PROGNAME",8
*2) Loading from cassette is done by LOAD"PROGNAME". Loading from disk by LOAD"0:PROGNAME",8
*3) Variable names in BASIC can be up to 16 characters long BUT ONLY THE FIRST 2 are recognised. Thus DATE and DAY will relate to the same variable. This can be the source of some errors in BASIC.
*4) Integer variable names have a '%' appended to the name not '#'.
*5) The DIM statement in BASIC is e.g. DIM A(5,6)
The index value always ranges from 0 to the number printed in the DIM statement. Negative indices are not allowed.
*6) To refer to parts of strings in BASIC one must use LEFT$, RIGHT$ and MID$. Refer to your BASIC manual for their definition. The examples in the text become:-

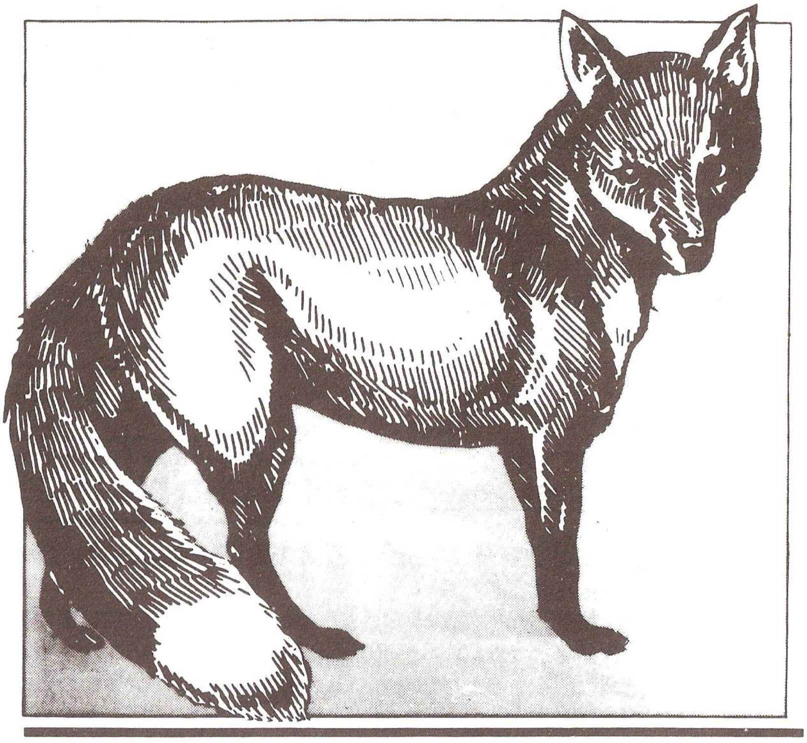
```
 A$=LEFT$(A$,4)+"12"+RIGHT$(A$,2)
 B$=LEFT$(A$,4)+"12"+RIGHT$(A$,2)
```

To be continued...

--oOo--

FOX ELECTRONICS

FOX ELECTRONICS



FOX ELECTRONICS

FOX ELECTRONICS 141, Abbey Road, Basingstoke, Hants. RG21 9ED

TEL: BASINGSTOKE 20671 (AFTER 6 P.M. - TIM OR JOAN)

ALL PRODUCTS ARE INCLUSIVE OF VAT. OVERSEAS CUSTOMERS PLEASE ADD £2.50 POST AND PACKING.

FULLY GUARANTEED FOR ONE YEAR.

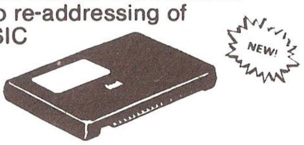
Deliveries 10 days from receipt of order.

FOX ELECTRONICS

FOX ELECTRONICS

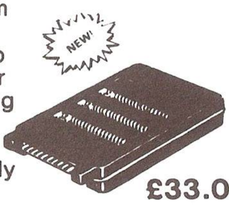
Products for the VIC 20

The VIXEN RAM CARTRIDGE.
for the Vic 20
Switchable between 16K or 8K & 3K. Gives you the option of full 16K RAM or 8K and 3K RAM in one package. When added to the standard Vic gives 16384 bytes of extra memory in blocks 1 and 2 or 3092 bytes of extra memory into the 3K memory block AND 8192 bytes switchable between memory blocks 1 and 3. Simply plugs into the rear expansion port and fully compatible with all motherboards and modules available. No re-addressing of existing BASIC programs needed.



£39.95

TANDEM
Expandable Expansion System, gives 4 expansion slots for Vic 20 cartridges. Custom designed case. Plugs directly into computer. Further expanded by using Tandem System ROM socket. No extra power supply needed.



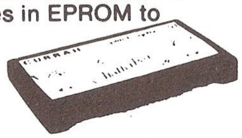
£33.00

VIC LIGHT PEN
A high quality light pen which plugs straight into your Vic with no special interface needed.

£19.50

or for PET 12" Screen £22.50

CHATTERBOX. Speech synthesizer with an infinate vocabulary of spoken words out of a number of sound units. Fully programmable and simply plugs into Vic or motherboard. Includes a series of software routines in EPROM to facilitate the programming.



£57.00

Please send me ....

ITEM	QUANTITY	PRICE	TOTAL

Name ....
Address ....
....

CLUB DISCOUNT - £1.00per item.   No P & P

DISK FILE - SECTOR 7

By Mike Todd

OOPS!

Well, it's "egg-on-face" time. The fix I gave on page 347 of Sector 7 for reading corrupted 1540/1541 disks on the 4040 was incorrect. The correct command is:

```
PRINT#15,"M-W";CHR$(92);CHR$(52);CHR$(1);CHR$(31);
```

In fact the location to be change is $435C and I miscalculated the CHR$ values and was changing location $34C5! Easily done when you're in a hurry - but that's no excuse. Also 217 in the first command on p347 should read 157.

DISK REVEALED

I have been promising for some time now to give a more detailed review of DISK REVEALED, but time and space are now getting so precious that I can't really go into any great discussions on it.

Anyway, let me tell you a little bit about it. It's written by ICPUG member Nigel Richman, and available from SUPERSOFT at £25.

It allows you to get at the data in individual sectors on the disk by simply specifying which track and sector you wish to see; the display comes out in hex format with the ASCII equivalents alongside.

On a 40-column PET, only half the sector can be displayed and so a single keystroke will switch you back and forwards between the lower and higher halves. Any data that is not part of the file (in other words data left over in the sector after the end of the file) is shown in reverse field to highlight it.

Having examined a specific block, it is very easy to alter it and then re-write it to the disk, although nothing will happen if a DOS MISMATCH or ID error occurs that has not been corrected.

Having read a single sector, you can continue along the chain of the file if the program is in "link mode". If you are in "track mode" you can examine consecutive sectors on the same track. In fact moving around the disk, just "browsing" is extremely easy.

If you are using the link mode to examine a complete data or program file, it is often useful to be able to step back through the sectors that you've already examined, and this has been thought of. The last (referred to as OLD) sector number is displayed on the prompt line at the top of the screen and continual pressing of "o" on the keyboard will step back through the file. Unfortunately the program is not clever enough to actually step back through a file that you haven't already stepped forward through, but such a task is virtually impossible, so I'll forgive this omission.

Pressing "e" will take you right through to the end of the file, and a copy of this is also kept to allow stepping backwards.

The screen also displays the current track and sector number, the next sector in the chain as well as other information such as the true ID of the disk (NOT the ID held in the directory, but the ID actually written at the start of every sector).

A HUNT facility is provided so that a file chain can be examined for a given hex or ASCII sequence, and the search is quick and versatile with "wild cards" being accepted such that the hex hunt on "20 ?? A?" will find every occurrence of a JSR $Axxx.

Certain disk errors can be recovered using the program, and on some occasions it is essential to avoid initialising the disk. On entry into the program there is the option of having the disk initialise or not. This feature should be used with care, as an uninitialised disk could easily get corrupted under certain circumstances.

The program works extremely well, and is very fast in operation. Much faster than some of the BASIC equivalents that I've seen, and more versatile. It does require a reasonably good understanding of the layout of the disk to use it to its fullest, but the program could be a useful learning tool in finding your way around the disk. But it doesn't provide the background knowledge which will have to be gathered elsewhere I'm afraid.

I would have liked to have seen a couple of other features in the package, such as the ability to read/write blocks of memory within the disk (including the "hidden" FDC memory), perhaps in a SUPERMON style.

I would also have like to have seen some disk logging features added that would allow the directory to be searched for a given file (or any combination of files) and give a listing of their starting track & sector, length, file type and perhaps even a dump of the first couple of bytes in each file. Also, perhaps a similar listing but done on a sector-by-sector basis - this could be accompanied by a "status" report of each sector showing any errors if found and also if the sector had or had not been used and was or was not included in the BAM.

But these comments should not be taken as a criticism of a useful program which, frustratingly, will not allow you to look at itself.

For anyone taking disks reasonably seriously, I would consider it £25 well spent, and there are versions for most drives and computers, although the 1541 version is slightly more restricted in its facilities.

--oOo--

DISCOUNT CORNER

For discounts on CBM Hardware and Software please write to me enclosing a stamped self addressed envelope for my reply and source (discount rates can change between Newsletter issues). Stocks are usually kept of 5.25" double density disks. These can be used as single- or double-density disks and some people have been successful in using them as double-sided (responsibility is not accepted for use as double-sided disks). Todays price is £ 16.60 for ten (minimum quantity ten disks or multiples of ten) and you should make out your cheque to ICPUG.

Please send a jiffy bag with your order already addressed (to yourself) and stamped to cover the weight of your jiffy bag plus 300 grams for ten disks. If you require twenty disks please adjust the size of the bag and the postage. The size of the box containing ten disks is six by six by one and five eighths inches (15 x 15 x 4 cms) and the jiffy bag has to be long enough to fold over with this box inside and wide enough to take the thickness (a number one jiffy type bag is not wide enough).

Special prices are obtainable for EPSON printers and interfaces, contact me for the source.

Superscript and Superspell are available through me with cheques made out to Precision Software Limited. Price each including VAT is £ 86.25 which now includes a backup disk and a manual. Superspell can be added to your existing Superscript for £ 86.25. Please send a stamp for forwarding your order to the Software House and state your disk format e.g. 4040 or 8050 (Superscript & Superspell are not available for Vic-20 or the 64). Easyscript for the 64 is CBM Software - please see the first paragraph.

Vizawrite and Vizaspell (for the 64) with manuals but not backup disks are available through me at the special prices of £ 46.00 and £ 36.00 respectively. Your cheques should be made out to Viza Software and you should include a stamp for forwarding your order to the Software House.

Wordpower for the PET only (not Vic-20 nor 64 - nor 700) can be obtained directly from the Software House by quoting your membership number at the special price of £ 35.00 (write to Kevin Pretorius, 15, The Vineries, Reservoir Road, Oakwood, London N14 4BH). 40- to 80-column conversions can be obtained from both Delph Electronics Limited, 4, Deeping Road, Baston, Peterborough, PE6 9NP ('phone 07786 535) and from Microport, 7, Clydesdale Close, Borehamwood, Herts WD6 2SD ('phone 01 953 8385) at 15% discount to members who should apply direct to the suppliers quoting their membership number (and any product query they may have).

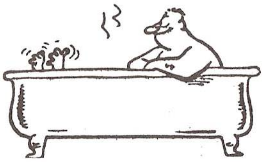
Cleveland Interface produce a Domalarm interface for the Vic-20 at £ 19.95 as a kit or £ 24.95 ready built and they will supply directly to members who quote their membership number at discounts of 12.5% & 10% respectively. The address is 18, Chelmsford Avenue, Fairfield, Stockton-on-Tees, Cleveland and any queries should be made directly to Cleveland Interfaces 'phone 0642 582626. If there is sufficient support we will be able to offer good discounts on items sold be Simple Software Ltd., (please peruse the usual Micro Mags for details or call them on 0273 504879) but the discount will depend upon the numbers of your orders through me and how long you are prepared to wait for your Software. Please write to me enclosing a stamped addressed envelope advising your order details and how long you are prepared to wait for your order to be filled. If you are patient enough you will find it very worthwhile!

JCL Software will allow members 15% discount on their IEEE Bus Adaptor for the Vic-20 and 64 which normally sells for £ 67.85 (club price £ 57.67). The same discount applies to their ROM Pager Boards, 8-way £ 51.75 (club price £ 44.00) 2 x 4 way £ 54.63 (club price £ 46.43) and to their EPROM Programmer Mk.3 £ 343.85 (club price £ 292.27) - new Fast Card for 4K & 8K EPROMS (2764 & 27128) available soon at a price soon to be announced. Members should apply directly to JCL Software for these items quoting their membership number. Address is 47, London Road, Southborough, Tunbridge Wells, Kent 'phone 0892 27454. The Business ROM for the 4000/8000 PETS is now £ 92.00 (for special club price please apply to me with stamped addressed envelope). The Business ROM is soon to be made available for the 700 in a cartridge. This cartridge has three sockets for 8K RAM or ROM chips and will be made available separately. Both items will be available to ICPUG members at special prices, please write to me for details with a stamped addressed envelope.

Sad news. The UK office of Solidus International has been closed down and all documents returned to B.C. Canada. This means that SYSRES is no longer available to ICPUG members.

Please do not telephone me during the day, weekdays, I will accept your calls after 8p.m. when I should have finished my evening meal.

J.B.



--oOo--

COMMODORE COLUMN

On August 1st new prices came into effect as Commodore axed the price of the C-64 to £ 229.00, inclusive of VAT, and for good measure dropped the 1541 disk drive to the same price. The 3K RAM Cartridge (VIC1210) is now £ 19.95, the 8K (VIC1110) £ 29.95 and the 16K (VIC1111) £ 39.95. The cartridge games, VIC1901 to VIC1927 and C64601 to C64606, are now £ 9.99. All these prices include VAT and may be less at certain stores.

A group of top Commodore dealers have formed a dealer co-operative to help each other in marketing and advertising Commodore products. An advertising agency has been appointed to handle the £ 500 per month which each dealer contributes.

R.D.G.

--oOo--

THE 64 COLUMN

By Mike Todd

64 CHARACTER SET

The character set tables I put in the May Newsletter actually have an error in the CHR$ tables, and I've given details of this in the VIC-COLUMN.

APPLICATIONS

Last time I mentioned that there were several application packages available for the 64, and I've had a couple of letters asking what the amateur radio package was all about. Although I've not actually seen it, it emanates from COMPUTER WORLD in Holland and for £139 you get a single board that plugs into the 64 and, provided that you hook up a separate mains transformer and connection to the user port (transformer, user port connector and cable are available at extra cost), you can have RTTY, CW and slow-scan TV operating on the 64.

It seems to do just about everything that you could possibly want. The screen is divided into several areas giving status displays and two text areas (one for incoming, the other outgoing).

Other features include: baud rates 45, 50, 75, 110 & 330; TX tones 800; 980/1180; 1650/1850; 1200/2400 (Kansas city interface); 1275/1445/1925; 2125/2295/2550/2975; ASCII or BAUDOT; CW at 5-99 wpm; text editing during RX mode; 12K text buffer; disk or cassette storage of text; disk mailbox facility; CW FSK ID for RTTY; page mode for RTTY pictures; QTR from internal 24-hour clock; printer on/off facility for printing incoming text; byte mode for sending program files; SSTV pictures on 100x120 dot format with four step greyscale; simple wordprocessing; paddle key facility; optocoupled input. That's enough to keep anyone going !

64 SOUND

As many of you probably know, my "proper" job (when I'm not writing for ICPUG etc.) is actually nothing to do with computers - I work in radio broadcasting as a sound engineer (well, sort of) - and I was therefore very interested to have a listen to the 64's sound on a good audio system. Although some of the sounds were really quite remarkable at "hi-fi" quality, I was very disappointed in the level of background noise generated by the 64, and would suggest that this will limit its use in more serious applications.

For those who are interested, I measured the noise separation as -44.5dB (referenced to 0dB=1mW into 600 ohms) with all the voices off and minimum volume (0), and this deteriorated to -25.5dB with all voices off, but volume at maximum (15). Putting the low pass filter (which substantially filters out the hf "hash") into circuit made about 6dB improvement, but then it also attenuated the audio signal.

I also tried some experiments feeding an external audio signal into the external input to the SID chip and was surprised how good the quality was on the output. A quick frequency response measurement showed that the response was within 1dB from 50Hz-25kHz, although there was some distortion present, especially when the filter was used.

Investigating the ADSR envelope more closely, I was dismayed to discover that the decay, especially at slow rates, was far from linear and as it approached the lower end, the level would often jump suddenly by 2-3dBs and then drop back again. I don't know why this should be, and can only suggest that it's a design problem with the SID chip.

Overall, I was quite impressed with the performance of the sound side of the 64, and some external filtering may manage to reduce the rather intrusive noise figures. The output level of all voices was remarkably constant across the full frequency range and this suggests that the 64 could be used as a programmable sweep generator for audio testing, as long as the noise figures are born in mind.

It is worth noting that appendix E of the Programmers Reference Guide (generating notes) is based on the American clock frquency of 1.022MHz and will result in the notes being significantly flat on UK machines with a clock frequency of 0.985MHz. For instance A=440 Hz actually comes out as 424Hz. For many applications this should cause no problems, but if you want to use the 64 to "play" along with a piano for instance, you will need to adjust the notes in the table to A=440 Hz. This means multiplying all the given values by 1.0382, which gives A-4 (which is the usual reference of 440 Hz) as 7493 and not 7217.

SIMONS BASIC or "NEVER MIND THE QUALITY, FEEL THE WIDTH"

If SIMON's BASIC (I'll refer to it as SiB from now on) was to enter a contest for the number of extra commands it makes available, I'm sure its total of 114 would put it amongst the front runners. But if it wins on quantity, it must surely be ridiculed when it comes to quality - for it has very little! First, let me briefly describe the commands without comment and then I'll present a few of the findings of some ICPUG members who have written to me with their comments on SiB.

SIMON's BASIC comes as a plug-in cartridge, and when powered up, has a grey screen, dark blue border and black characters and with 30719 bytes available, the rest are used by the cartridge. Immediately, the SiB commands are available.

The first group of commands are the programming aids which include the ability to assign character strings to each of the function keys so that when one of the keys is pressed, the programmed characters are printed on the screen. There's also a DISPLAY command which will list all the keys and their assigned strings.

AUTO gives automatic line numbering and RENUMBER will renumber all lines in a program. PAUSE halts program execution for a specified number of seconds, pressing the RETURN key aborts the pause to allow the program to continue.

CGOTO (Computed GOTO) allows an expression in place of the normal line number, RESET allows READ commands to start reading data at a specified line number and MERGE will load a previously saved program and tag it onto the end of a program already in memory.

There are a couple of aids used when listing programs, for instance the listing can be halted every few lines by using the PAGE command, DELAY will slow the listing down when the CBM key is held down and OPTION makes any of the SiB keywords stand out in reverse field.

There is a FIND command, to find specified characters in a program and a TRACE command which prints the line numbers that the program is executing in a little window at the top right of the screen. All program variables (except arrays) can be listed using DUMP.

Program lines can be "hidden" from gaze when listed and a complete reset of the 64 can be achieved with the COLD command. Any program that was in the 64 prior to the 64 being reset can be recovered using the OLD command, as can a program that was accidentally NEW'd.

INSERT is one of the text manipulation commands and allows one string to be inserted into another at a specific point, INST does the same except that the substring will overwrite the main string. DUP creates a repetitive string and PLACE finds a string within a string and returns the position.

Printing AT a specific position on the screen is available, strings can be printed in the CENTRE of the screen and there is a simple PRINT USING command for numbers held as strings.

Input is not forgotten and a limit to the range of characters accepted can be imposed by the FETCH command, while INKEY returns the value of a function key, if one is pressed.

If during your program you want to jump to a specific line as soon as a given key is pressed, then ON KEY is available to do this and it has two companion commands, DISABLE and RESUME.

Extra numeric functions are available too. MOD(X,Y) returns the remainder when X is divided by Y while DIV(X,Y) returns the integer part of the same operation and FRAC(X) the fractional part of the number X. EXOR(X,Y) is a logical function which returns the result of X exclusive OR'd with Y. Non-decimal numbers can be used in expressions by preceding those in binary with %, and hex numbers with $.

DISK will send a command string to the disk and DIR"$" will print the directory on the screen, without loading it.

SiB allows some structured commands such as IF...THEN...ELSE, REPEAT...UNTIL and LOOP...END LOOP with a facility to EXIT the loop under specified conditiions. Also, if you need to repeat the test in an IF..THEN line, RCOMP does it.

Procedures can be named and executed or even jumped to, and variables within a procedure can be defined to operate locally until the GLOBAL command is executed, when all local variables return to their global values.

To help programmers cope with errors, there's an ONERROR command with ERRN and ERRLN variables to return the error number and line number where the error occurred.

A variety of commands allow reading of joysticks, paddles and light pens as well as providing some elementary music commands to define a voice's ENVELOPE and VOLume, then set up a string of MUSIC and PLAY it on one voice.

As far as the screen is concerned, this is where SiB has its greatest number of commands available. High resolution commands such as PLOT, LINE, CIRCLE, ARC, ANGL, REC, BLOCK, DRAW and PAINT are available and these plot a point, line, circle, arc, a circle radius, a rectangle, block and arbitrary shape defined as a string (a little like turtle graphics) and finally fill any enclosed area with a specified colour. The screen co-ordinates start at 0,0 at top left to 319,199 in bottom right, unless you're working in MULTIcolour, in which case the bottom right is limited to 159,199.

TEXT and other CHARs can be positioned on a graphics screen, and in a variety of sizes, while shapes defined in strings and drawn using the DRAW command can be scaled up and rotated through multiples of 45 degrees.

The screen can be FLASHed (BFLASH flashes the border) or scrolled left, right, up and down, saved or loaded from disk or cassette, copied to the printer, rearranged in blocks and areas filled with specific characters.

Sprites (or rather MOBs) are also not forgotten - they can be designed (as, incidentally, can the character set), turned on and off, positioned or even moved about by SiB (one at a time). There are also DETECT and CHECK commands for detecting sprite/sprite or sprite/character collisions.

And that's the lot, apart from two final commands that are mentioned in the introduction to the handbook but not described anywhere. LIN will return the current line number of the cursor (just like POS returns the horizontal position) and SOUND is a reserved variable which is the start address of the sound chip (SOUND=54272).

The handbook is large, spiral bound and runs to about 150 pages. It has several example programs illustrating the use of some of the 114 SiB commands.

WHAT ABOUT THE QUALITY?

My overall impression is one of a program written by someone who has little experience of the art of writing utility packages and with a very limited experience of other systems.

Many things have been overlooked, for instance null strings are rarely catered for and an attempt to clear a KEY definition using KEY 1,"" will put garbage into the key. It has been known for this garbage to contain other KEY strings, which could be problematical if this were to contain NEW or something similar.

Changing the program text into tokens is handled sloppily. For instance SiB keywords in DATA statements are changed into tokens and will read as garbage so that TROUT is set up as TR+OUT, with OUT being tokenised and will not be read as TROUT, even though one of an example program in the handbook has such DATA lines in it.

Any keywords which appear in hex numbers (e.g. $CDEF) are also tokenised so that an attempt to PRINT $CDEF or $DEFC will result in an error message as DEF is tokenised and is not a hex character.

While on the subject of hex numbers, the handbook doesn't tell you that hex numbers must contain 4, and only 4 characters, and binary numbers must contain exactly 8 characters. There's also no way of converting numbers into hex or binary.

RCOMP doesn't, as the text says, repeat the condition test, it simply repeats the result of the previous IF condition so that, if the conditions are changed, they must be retested.

SiB keywords cannot be abbreviated in the same way as those in normal BASIC and, as some of the commands are quite long, this can become a little tedious. Frustration is further increased by the fact that spaces between keywords and their parameters are sometimes essential and sometimes they must not be present, although the handbook insists that all SiB keywords must be separated by a space.

Many command parameters are not checked to see if they are within range, and this can result in some very odd effects - for instance DUP("XXX",100) would theoretically result in an illegal string, but this is not checked and the resulting string seems to be something of an arbitrary length.

RENUMBER only renumbers the program lines, and ignores GOTO, GOSUB and so on. This may be a means of forcing the user to use structured programming (there's no need for GOTO or GOSUB when using procedures). But then why does SiB have a couple of its own specialised GOTO commands (ON ERROR and ON KEY)?

If the MEM command is used to move the character ROM into RAM, the screen moves to 52224 ($CC00) but some of the routines which write directly into screen RAM (like TRACE) don't take any notice of this and continue to write into the original screen. This could be disastrous if this area were being used for something else.

The additional arithmetic commands only work in integers in the range 0 to 65535. For EXOR, this is odd as it is different to the other logical functions (AND, OR etc) which work in the range -32767 to +32767.

One thing that I found particularly irritating was that the ability to flip from text to graphics and vice versa using either the CBM+SHIFT keys or CHR$(8) and CHR$(9) was locked out.

These then are some of the bugs in the package, there are also many errors in the handbook. There is a distinct feeling here that SiB was originally written for the VIC as some VIC-type statements appear. The colours in the table on p6-2 are actually those of the VIC and not the 64 and on p7-9 the scrolling limits are given as 23 characters by 24 lines which is nearer the VIC's screen than the actual limits of 40 characters by 25 lines.

There are also some straightforward typing errors, such as the COLOUR command whose format is COLOUR bo,sc and not, as p6-3 shows, COLOUR sc,bo and REPEAT needs a colon following it.

The handbook has many program examples, some of which do not work. The sprite demonstration program on page 8-4, for instance, sets the sprite data starting at location 2048 which is where the BASIC program will be !

Apart from these mistakes of type setting and programming, there are many places where SiB just isn't what it ought to be. These inconsistencies and poor attention to detail indicate that the package was written in a piecemeal fashion without any idea of the end result. It is vital when preparing a language extension of this sort that a great deal of thought goes into specifying the final product before any coding ever takes place.

For instance, MERGE isn't a merge at all, but a simple APPEND command - why is it not named as such ?

TRACE uses white characters on the grey background - almost impossible to read. TRACE doesn't show the line currently executing, it only shows the line number when the NEXT line is started.
CGOTO is a separate keyword, why not just preserve the original GOTO and GOSUB and add the computational bit ?
FIND will not find sections of strings in quotes, for isntance FIND "AB" will not find "ABCDE" but will find "AB", and only the line numbers are printed out, not the complete lines as in other programmers aids.
FCHR fills a screen area with a specified character, but doesn't set up the colour memory - so you can't see them!

PAGE splits the listing up into "pages" of a given number of lines - if you list in this way, and then want to modify a line you've seen, you cannot abort the list and must procede to the end of the program. Boring!!

The arithmetic functions use a slightly odd format - why not use A=X DIV Y or B=C EXOR D and keep consistency with other arithmetic operators ?

Some options in SiB are selected, not by the words ON or OFF, but by using numbers. For instance TRACE is turned on by TRACE 10 and off by TRACE 0 !
RLOCMOB moves a specified sprite across the screen at a specified rate - but only one at a time.
BFLASH and FLASH both affect the interrupt routines and as a result the TI variable is slowed down significantly.

The normal use of the RESTORE key is to set everything back to normal - but flashing borders and characters remain.

Plotting has the origin at the top left of the screen whereas most plotting is required with the origin at the bottom left. Also, the range of co-ordinates is different for normal high resolution and multicolour. I know that this is reasonably logical, but it makes it difficult to set up a screen in high-res and then see what it would look like in multi-colour. This is also made difficult by the incompatibility of the plot type codes between the two modes.

Incidentally, none of the graphics commands are compatible with the VIC's SUPER EXPANDER and so it is going to be very difficult to transfer programs across, despite the fact that the SUPER-EXPANDER was written with such portability in mind.

The idea of a DISK command to send commands directly to the disk without the need to OPEN a command channel is very good, but why is there no way of reading the command channel for error messages? Also, why does the DIR command have to have the "$" after it, surely DIR on its own should be allowed ?

The USE command for print formatting is very useful, except I can't understand why a numeric variable shouldn't be allowed instead of only a string.

The use of procedures in a program is a good way of structuring a program, but a procedure should really allow parameters to be passed to it, and there should also be explicit local variables. SiB uses procedures as nothing more than labelled line numbers - and EXEC acts like a GOSUB, CALL like a GOTO and END PROC like a RETURN.

LOCAL allows variables to be temporarily deleted from the variable table, so that further use of them will make new entries (although it doesn't work properly for arrays). GLOBAL re-instates all the "hidden" variables, but actually leaves the redefined variables in the table.

Therefore:

```
        A=5 :LOCAL A :A=10 :PRINT A :GLOBAL :PRINT A
```

will produce the output of 10 and then 5, but a DUMP will show two values for A. The first is 5 and the second 10. However, this last value will not normally be seen by BASIC as it is the first occurrence of a variable in the table that is accepted.

Sib does have some useful features. Its graphics commands will save a lot of PEEKing and POKEing and the structured programming commands will help newcomers to the art, but many of the facilities are rather difficult to use and the bad points really do outweigh the good. I cannot understand why Commodore ever agreed to go with the package, unless the excitement and publicity of a teenage boy hero was too much for them to resist.

I don't want to seem too negative about it, but it's very difficult, and my advice would be to avoid it. There will be other packages from other authors, and although they may not have so many commands available, would be worth waiting for.

At least they kindly gave the author's address in the handbook so that disgruntled users can pass their comments direct to the author himself - but Commodore's contact is also given, so maybe comments to both would be better !

Finally, I would like to thank the many ICPUG members who have contributed their comments to this "review" - especially Eric Davies and Mike Rose both of whom had some good things to say about it ! But, over all the comments I received, the bad points outnumbered the good by 10 to 1.

--oOo--

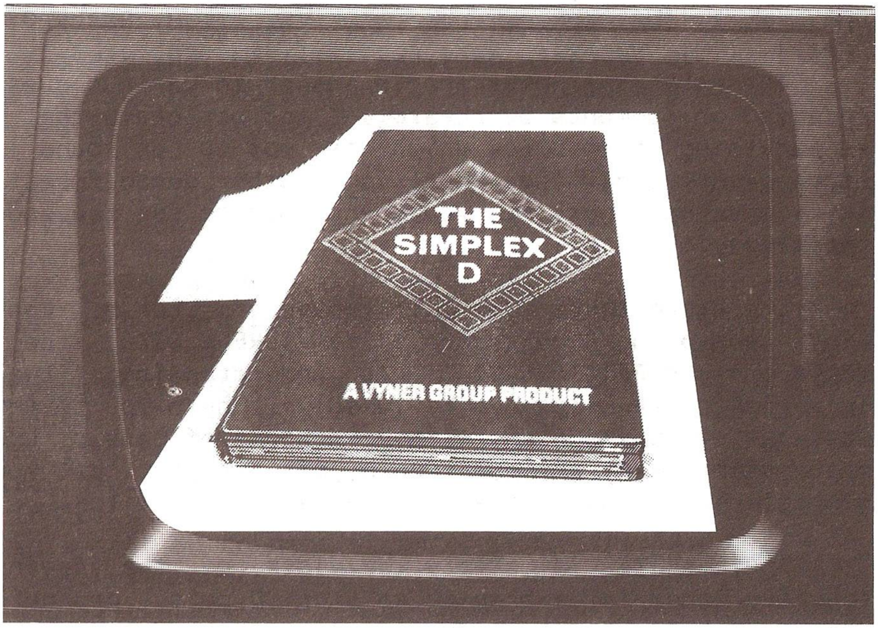
CAVEAT EMPTOR

Whilst rumaging around for Commodore-related books in a London bookshop, I came across a fascinating book entitled "The VIC-20 Computer Program Writing Handbook" - by Howard Adler and published by ARCSOFT (Maryland) at £4.25.

It certainly looked an interesting book from the outside, with apparently 92 pages of program writing tips. But, opening up the book presented a very different picture. The first ten pages were certainly of some use, describing the BASIC commands available. The next 80 consisted entirely of program coding forms. You know, the forms covered in little boxes to make writing programs easier, and really of little use unless you're writing in FORTRAN for use with punched cards. The last two pages were "graphics grids" to help in designing screen layouts.

If you actually have a chance to look at the book before you buy it, I'm sure that you'd soon put it back on the shelf again. But if you see it in a mail order ad, then you could quite easily be taken in. If you see it advertised, don't be tempted. At nearly 50p for each page of information it's just not worth it!

--oOo--



The Electronic Cash Book

Micro-Simplex makes Retailers Accounts and Stock Control simple...

Unique features:

- Based on Britain's No. 1 cash book system
- Uses Britain's No. 1 business micro computer
- The only one recommended by Vyners, publishers of Simplex books
- The only one offering all retailers special V.A.T. schemes

Other features include . . .

- Stock control linked to cash registers
- Simple and familiar layouts
- Easy to use
- Automatically produces:
  - (a) Statements to customers
  - (b) Lists of unpaid bills
  - (c) Simple profit and loss accounts

MICRO-SIMPLEX

MICRO - SIMPLEX

OFFICIALLY APPROVED BY COMMODORE

commodore COMPUTER

Contact any Commodore Dealer

or

Micro-Simplex Limited,

8, Charlotte Street West,

Macclesfield, Chesire.

Tel: 0625 615000

THE SUPERSCRIPT SCREEN

By Barry Biddles

The following notes are in answer to various questions received or comments overheard. I do not claim that there is anything given below which is not to be found in the manual. However, the manual is a rather dense forest, and it seems that some of you are unable to see the wood for the trees.

There are three important parameters to be kept in mind when using Superscript, the SCREEN SIZE, the TEXT WIDTH and the PRINTING SIZE. An understanding of the ways in which these sizes can be used will help you to get the most out of Superscript, either in terms of ease of use, or maybe simply the maximum length of document that you can generate in one piece, before having to link files.

SCREEN SIZE is simply the width of the screen, in columns, and is either 40 cols or 80 cols, depending on which model of computer you are using.

TEXT WIDTH is the width in columns of the text that you key in to the computer. When you first set-up Superscript, immediately after loading the program, if you press <RETURN> in response to the first question, Superscript will recognise which machine you are using and set the text width to the same value as the screen size. However, for special applications, you may enter any number equal to or greater than that, up to a limit of 240. The screen now behaves as a small window through which you can see the much wider text. The window moves up and down, and even sideways, to enable any portion of this text to be brought into view._

PRINTING SIZE has nothing whatever to do with either of the above. Re-read that sentence slowly. There are MARGIN commands, which may be placed anywhere in the text, and which set the positions of the left and right margins ON THE PAPER. If you do not set margins, they default to

Left Margin 1 and Right Margin 80 (or Text Width if this is wider than 80). However, you may choose any values you like, provided that Left is less than Right by at least 15, and Right is no more than 203 (the maximum number of characters at 15-pitch on a Spinwriter).

Superscript has a wonderful facility called Output To Video, which should usually be used prior to Output To Printer, to check that all is well. The text is printed onto the screen exactly as it would be on the paper, complete with margins, so that you can see whether everything is in the right place, and whether the end-of-page breaks occur in convenient places, etc. If your paper width is the same as the screen width (in columns), then you will see the final layout just as on the paper. This will happen, for example, if you are printing to A4 paper (80 cols of 10 pitch text) and using an 8032, which can display all 80 columns at once. However, if you try to do the same on a 40-col machine, the 80-col lines will not fit, and will have to 'wrap around' onto the next line. With a little practice, you can see easily enough where the left and right margins are, even though they are on alternate lines.

It is slightly tedious to read a lot of text if the text width is greater than the screen width, because the screen keeps having to move, and you can't see the whole of the line at once. You would certainly not want to work like this the whole of the time. But for special occasions, this disadvantage is far outweighed by the ease of entering the table just as you want it. The Output To Video is still just as confusing, of course, but now less necessary because you have seen the final layout by this alternative means.

A small point, however - put a comment line at the top of the file.

```
     <tick>CM SET UP IN 61 COLUMNS
```

If you forget to set the text width correctly before LOADing, the text will look very strange, and it will take you quite a while to work out what text width you should have used. (To reset the text width, just stop the program with RVS CLR, which takes you back to the opening menu. Answer the questions the required way, and you will be back to EDIT MODE again, without loss of the old text).

--o0o--

HELP WANTED

Part of our Software Library service involves copying disks. The 8050 format disks have hitherto been replicated by Stephen Rabagliatti but Stephen can no longer continue this service. We are therefore seeking someone who is prepared to undertake this task. Offers of help to the Chairman please.

--o0o--

ROUND THE REGIONS

South-west Scotland region have acquired a new venue and as a result have changed their meeting dates. Meetings are now on alternate Thursdays, starting September 1st, from 7 to 9 p.m. in the Symington primary school, Brewlands Road, Symington, Kilmarnock.

At the June meeting of the Watford region members were treated to a demonstration of Superscript II for the CBM700. A hint of things to come emerged with a database package for the 64 and the 700.

North Hampshire region were fortunate to see the SX-64, the portable 64 with colour screen and floppy disk built-in. The demonstration program was not exactly seasonal, and I could not help but wonder if it was written with the coming, or last Christmas in mind ! Never-the-less the pictures and accompanying music were most impressive.

Watford Region meetings will in future be held at the Co-operative Hall in St. Albans Road, Watford.

--o0o--

Advertisement

LINKPLUS LIMITED - COMPUTERS IN ASTROLOGY

23 Park Hall Road London N2 9PT
Telephone 01-444 5104

We have a wide range of computer programs to create astrological data, charts and character reports. The size of these programs range from 3kb to around 200kb of software.

The programs run on the VIC, CBM-64, 3000, 4000 and 8000 series and, soon on the 700 series. They run on other micro's, but we won't mention those here!

We offer a ten page report on your basic character for 6.95 for a twelve page report on your reactions over the next three months for 6.95. Both reports cost 10.95. If you quote your current ICPUG membership, then deduct 2.00 off each report or 4.00 off the combined offer.

We would require your date and place of birth. Please show the month in letters, three will do, to avoid confusion. For those born in a village it would help to know the nearest town. For three month reports, give the starting date - it must be from the 1st of a month.

If you can state the time of your birth it will be helpful. If not, don't worry, we can use a solar chart which will only affect one chapter out of ten (i.e. ignore chapter 2, the other 9 will be correct).

As a SPECIAL OFFER to launch our advertising in ICPUG magazine, we will reduce the price of our character report, still further, to 2.95, if ordered before 1st September 83.

If you want to purchase the programs send a stamped addressed envelope for details or ring 01-444 5104 between 6.00pm and 8.00pm any evening or after 10.00am on weekends to take advantage of Buzby's cheep (sic) rate. Our programs start at under 20.00 but quite a few of the more sophisticated ones are extremely complex and cost many hundreds of pounds.

COMAL CORNER

My contribution to COMAL corner this time is going to be much smaller than usual. Too busy with my other articles! However I do have some news.

I have found a bug in revision 01 of COMAL64S for the C-64. Fortunately not many people have a copy of this version and anybody who responded to the offer of COMAL in the July Newsletter will have got revision 02 which solves the problem. In order to enable COMAL to operate with the latest Commodore printers I had to stop COMAL terminating its lines with CHR$(13), CHR$(10) by removing the CHR$(10). Unfortunately because of a bug in the original COMAL version 0.12 code this meant that files that were ENTERed were not properly closed. Files ENTERed from cassette did not stop at all. Revision 02 has solved the problem by ensuring an 'end of file' status will terminate the ENTER command. On version 0.12 for 4032 machines it was only the fact that reading a disk after the 'end of file' returned a CHR$(13) that enabled ENTER to terminate at all. Most inelegant. One interesting point raised by this problem is that I now understand what an 'end-of-file' error is. It is not that the end of file has been reached but that an attempt to read BEYOND the end of file has been made.

I neglected to mention last time the improvements made in the 0.64 version of COMAL.

(1) It works with cassette.

(2) It allows a pause during LIST by pressing the 'space' key. Any other key (except STOP!) restarts the listing.

(3) The pause facility also exists during the CAT command for reading disk directories.

(4) Two bugs in the AUTO command have been corrected. The AUTO command has been modified to form the next line number from the current screen line number plus the increment. This is useful if one wants to back up a few lines to

correct an error. Also line numbers greater than 9999 will now cause AUTO to terminate.

(5) If a disk is not connected to the system, error numbers will automatically be given rather than trying to find a non existent error message file. This means that version 0.64S can be used without disks and the disks can be turned off if needs be.

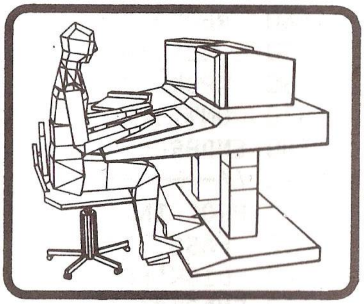
The other big plus with COMAL for the C-64 is that it is by far the easiest way to use relative files with the 1541 and the C-64.

Well that is my news for this month. Some time ago I modified a program to save from disk to tape which was published in the Newsletter. The original was written in BASIC by Geoff Pyke and Martin Simpson. Both of these programmers use COMAL and Geoff tells me he now does much of his programming in COMAL because it is easier to produce his programs this way. An example of Geoff's work follows. One of his disk utilities is an 'Unscratcher' which enables you to recover unintentionally scratched disk files, provided they have not been overwritten. Here is the program:

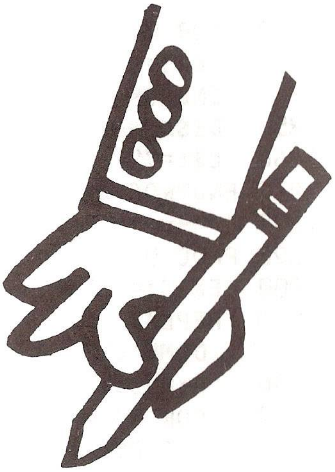
```
1000 // DISK PROGRAM UNSCRATCHER
1010 //
1020 // WRITTEN FOR 2031 DRIVE
1030 // USING DOS 2 AND 4032
1040 //
1050 // BY GEOFF PYKE
1060 //
1070 // MAY 17TH 1983
1075 // UPDATED 17/07/83 (VERSION 0.12)
1080 // ------------------------
1090 //
1091 POKE 115,126 // RESET END OF COMAL
1092 POKE 117,126 // TO $7EFF
1100 DIM DISKNAME$ OF 16
1110 DIM FILENAME$ OF 16
1120 DIM PRESS$ OF 24, YN$ OF 1
1130 DIM KEY$ OF 1, C$ OF 16, FT$ OF 3
```

```
1140 DIM GET$ OF 3, TT$ OF 3, SS$ OF 3
1150 DIM TR$ OF 3, SR$ OF 3, ST$ OF 2
1160 DIM CT$ OF 3, CS$ OF 3, ER$ OF 2
1170 //
1180 PRINT "<clr>",TAB(15),"<11'shift-$'>"
1190 PRINT TAB(15),"<rvs>UNSCRATCHER"
1200 PRINT TAB(15),"<11'shift-#'>"
1210 PRINT
1220 PRINT " THIS PROGRAM WILL UN-SCRATCH YOUR"
1230 PRINT "<dn> PREVIOUSLY SCRATCHED FILES PROVIDED"
1240 PRINT "<dn> THAT YOU HAVE NOT YET OVERWRITTEN THEM"
1250 PRINT "<39'shift-@'>"
1260 PRINT "<dn> PLACE YOUR DISK ON DRIVE 0 <dn>"
1270 //
1280 GETKEY("<3sp>")
1290 //
1300 PRINT "<up><39'shift-@'>"
1310 //
1320 INITIALISE
1330 //
1340 LOAD'MACHINE'CODE
1350 //
1360 OPEN'CHANNEL
1370 //
1380 READ'DISKNAME
1390 //
1400 PRINT "<clr>DISKNAME = : <rvs>",DISKNAME$
1410 //
1420 UNSCRATCH
1430 //
1440 CLOSE FILE 2
1450 //
1451 POKE 115,127 // RESET END OF COMAL
1452 POKE 117,127 // BACK TO $7FFF
1460 END
1470 //-------------------------------
1480 PROC GETKEY(GET$)
1490  B:=158; A:=623
1500  PRESS$:="<rvs>PRESS SPACE TO CONTINUE"
1510  POKE B,0
1520  POKE A,0
1530  IF GET$(1)=" " THEN PRINT TAB(9),PRESS$
```

```
1540  REPEAT
1550   KEY$:=CHR$(PEEK(623))
1560   POKE B,0
1570  UNTIL KEY$=GET$(1) OR KEY$=GET$(2) OR KEY$=GET$(3)
1580 ENDPROC GETKEY
1590 //-----------------------------
1600 PROC INITIALISE
1610  PASS "IO"
1620  DISK'ERROR
1630 ENDPROC INITIALISE
1640 //-----------------------------
1650 PROC DISK'ERROR CLOSED
1660  DIM ER$ OF 23
1670  ER$:=STATUS$
1680  IF ER$<>"00, OK,00,00" THEN
1690   PRINT TAB(11),"<rvs>",ER$
1700   STOP
1710  ENDIF
1720 ENDPROC DISK'ERROR
1730 //-----------------------------
1740 PROC OPEN'CHANNEL
1750  OPEN FILE 2,"#",WRITE
1760  DISK'ERROR
1770 ENDPROC OPEN'CHANNEL
1780 //-----------------------------
1790 PROC READ'DISKNAME
1800  DUMP'BLOCK("18","00")
1810  A:=32512
1820  FOR B:=A+144 TO A+161 DO
1830   C$:=C$+CHR$(PEEK(B))
1840  ENDFOR B
1850  DISKNAME$:=C$
1860  C$:=""
1870 ENDPROC READ'DISKNAME
1880 //-----------------------------
1890 PROC UNSCRATCH
1900  EXIT:=FALSE; TR$:="18"; SR$:=",1"
1910  REPEAT
1920   DUMP'BLOCK(TR$,SR$)
1930   S:=32514
1940   FOR SCAN:=1 TO 8 DO
1950    IF PEEK(S)=0 THEN RECLAIM
```

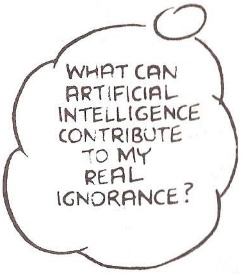


```
1960     S:=S+32
1970     IF EXIT=TRUE THEN SCAN:=8
1980    ENDFOR SCAN
1990    STR(PEEK(32512),ST$)
2000    TR$:=ST$
2010    STR(PEEK(32513),ST$)
2020    SR$:=","+ST$
2030   UNTIL TR$="00" OR EXIT=TRUE
2040  ENDPROC UNSCRATCH
2050  //-------------------------------
2060  PROC RECLAIM
2070   FOR NM:=S+3 TO S+18 DO
2080    C$:=C$+CHR$(PEEK(NM))
2090   ENDFOR NM
2095   IF C$(1)=CHR$(0) THEN
2096    C$:=""
2097   ELSE
2100    FILENAME$:=C$; C$:=""
2110    PRINT "<dn><rvs>",FILENAME$,TAB(20),"RECLAIM : ",
2120    GETKEY("YNX")
2130    CASE KEY$ OF
2140    WHEN "Y"
2150     PRINT KEY$
2155     NT:=S+1; RECL:=TRUE; CT$:=TR$; CS$:=SR$; F:=S; P:=1
2160     ALLOCATE
2170     IF RECL=TRUE THEN FILETYPE
2175     DUMP'BLOCK(CT$,CS$)
2176     TR$:=CT$; SR$:=CS$
2180    WHEN "X"
2190     EXIT:=TRUE
2200    WHEN "N"
2210     EXIT:=FALSE
2220     PRINT KEY$
2230    ENDCASE
2237   ENDIF
2240  ENDPROC RECLAIM
2250  //---------------------------------
2260  PROC ALLOCATE
2280   REPEAT
2290    STR(PEEK(NT),ST$)
2300    TR$:=ST$
2310    STR(PEEK(NT+1),ST$)
```

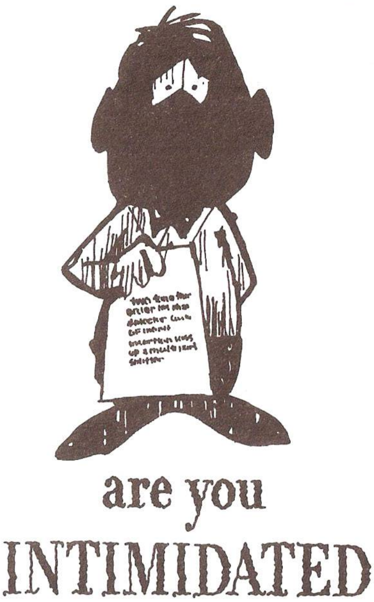


```
2320   SR$:=","+ST$
2325   IF P=1 THEN // FIRST PASS
2326    TT$:=TR$; SS$:=SR$; P:=0
2327   ENDIF
2330   IF PEEK(NT)<>0 THEN BLOCK'ALLOC(TR$,SR$)
2340   NT:=32512
2350   DUMP'BLOCK(TR$,SR$)
2360  UNTIL TR$="00" OR RECL=FALSE
2370 ENDPROC ALLOCATE
2380 //-----------------------------
2390 PROC FILETYPE
2400  DUMP'BLOCK(CT$,CS$)
2410  INPUT "ENTER FILETYPE (PRG/SEQ/REL): ": FT$
2420  IF FT$="PRG" THEN POKE S,130
2430  IF FT$="SEQ" THEN POKE S,129
2440  IF FT$="REL" THEN
2450   NT:=S+19
2460   ALLOCATE
2470   DUMP'BLOCK(CT$,CS$)
2480   IF RECL=TRUE THEN POKE S,132
2500  ENDIF
2540  IF RECL=TRUE THEN WRITE'BLOCK(CT$,CS$)
2550 ENDPROC FILETYPE
2560 //---------------------------------
2570 PROC BLOCK'ALLOC(TR$,SR$)
2580  PASS "B-A:0,"+TR$+SR$
2590  ER$:=STATUS$
2600  IF ER$="65" THEN
2610   PRINT "SORRY NOT POSSIBLE - FILE CORRUPTED"
2620   DEALLOCATE(TR$,SR$,TT$,SS$,ST$)
2625   FOR F:=S+3 TO S+18 DO
2630    POKE F,0 // DELETE FILENAME
2635   ENDFOR F
2640   WRITE'BLOCK(CT$,CS$)
2660   RECL:=FALSE
2670  ENDIF
2680 ENDPROC BLOCK'ALLOC
2700 //---------------------------------
2710 PROC DUMP'BLOCK(TR$,SR$) CLOSED
2720  POKE 1,0
2730  POKE 2,127
2740  PASS "U1:2,0,"+TR$+","+SR$
2750  PASS "B-P:2,0"
```

```
2760  DISK'ERROR
2770  SYS 674
2780 ENDPROC DUMP'BLOCK
2790 //--------------------------------
2800 PROC STR(A,REF ST$) CLOSED
2810  // CONVERT INTEGER OF LESS THAN
2820  // 100 INTO A 2 BYTE STRING
2830  ST$:=CHR$(INT(A/10)+48)+CHR$((A MOD 10)+48)
2840 ENDPROC STR
2850 //--------------------------------
2860 PROC WRITE'BLOCK(TR$,SR$) CLOSED
2910  POKE 1,0
2920  POKE 2,127
2930  PASS "B-P:2,0"
2990  SYS 698
3000  PASS "U2:2,0,"+TR$+","+SR$
3010  DISK'ERROR
3020 ENDPROC WRITE'BLOCK
3030 //--------------------------------
3040 PROC LOAD'MACHINE'CODE CLOSED
3050  // SET UP MACHINE CODE DISK
3060  // READ ROUTINES INTO 2ND
3070  // CASSETTE BUFFER.
3080  // SET WORK AREA POINTERS TO
3090  // $7F00
3100  POKE 1,0
3110  POKE 2,127
3120  X:=674
3130  REPEAT
3140   READ XX
3150   POKE X,XX
3160   X:=X+1
3170  UNTIL EOD
3180  DATA 162,2,32,198,255,160,0,32,207
3190  DATA 255,145,1,200,240,2,208,246,230
3200  DATA 2,76,204,255,0,0,162,2,32,201
3210  DATA 255,160,0,177,1,32,210,255,200
3220  DATA 240,2,208,246,230,2,76,204,255,0,0
3230 ENDPROC LOAD'MACHINE'CODE
3240 //--------------------------------
3250 PROC DEALLOCATE(TR$,SR$,TT$,SS$,ST$) CLOSED
```


WHAT CAN ARTIFICIAL INTELLIGENCE CONTRIBUTE TO MY REAL IGNORANCE?

```
3255  DEALLOC:=FALSE
3265  REPEAT
3275   IF TT$=TR$ AND SS$=SR$ THEN
3276    DEALLOC:=TRUE
3278   ELSE
3279    PASS "B-F:0,"+TT$+SS$
3280    DUMP'BLOCK(TT$,SS$)
3285    DEALLOC:=FALSE
3290   ENDIF
3295   STR(PEEK(32512),ST$)
3300   TT$:=ST$
3305   STR(PEEK(32513),ST$)
3310   SS$:=","+ST$
3315  UNTIL DEALLOC=TRUE
3320 ENDPROC DEALLOCATE
```



--oOo--

COMPUTER AID WANTED

S.J.Branson, 111, Park Road, Peterborough, is an experienced machine code programmer who has an interest in the computer aided layout of printed circuit boards. He would like to contact anyone with a detailed knowledge of the algorithms used in order to develop jointly a programming package.

--oOo--

MODEM ON A CHIP

Several members have written to me regarding modem construction and one region is purported to have a modem project on the go (but they haven't contacted me) and now semiconductor manufacturer Advanced Micro Devices have produced a 'World-chip' modem. This device comes in a 28-pin package and can be programmed to nearly 20 different modem standards. Data may be obtained from AMD (UK) Ltd., AMD House, Goldsworth Road, Woking, Surrey, GU21 1JT. Tel: Woking (048 62) 22121. Price around £ 40.

R.D.G.

--oOo--

Compuprint Computers Ltd

announce

'CAPS'

A NEW CONCEPT IN ACCOUNTING WITH SMALL COMPUTERS

This new program is available for all Commodore Machines (with minimum of single disc drive) from VIC20 to 8096.

It is a fully integrated sales/purchase/nominal package which does everything from simple bookeeping to full Profit and Loss Account and Balance Sheet.

The product of over a years research, these programs, used by our own computer bureau bring truly sophisticated accounting to the small computer.

One feature of particular note is that all information is stored, for ever, e.g.invoices are not deleted when paid. This allows for better analysis of data and more important, full audit facilities. Despite this, even single disc versions have capacity for several hundred accounts.

At £245 for VIC20/C64 and £395 for PET (ex VAT) the benefit of these programs can offset the cost in weeks.

For further details see enclosed leaflet or contact Compuprint Computers Limited.

INTRODUCTORY OFFER- see the full CAPS system in the CAPS manual. £20, refundable on full purchase

(DEALER ENQUIRIES WELCOME)

MAIL LIST

A sophisticated mail list program for all Commodore machines with disc drives.

All usual facilities plus -
- holds telephone, contact, postcode as well as name + 5 line address
- user definable label size and web width
- five separate code letters for selection in various combinations while printing
- sophisticated sort facility (even on single disc) which will sort to alphabetic order by any field or combination of fields
- sort facility sorts by surname without having to enter backwards (i.e. Smith W)
- hold over 1000 records on 1541/4000 disc, upto 6000 on 8250 format
- interfaces with CAPS accounting program

PRICE TO ICPUG MEMBERS £99 inc.VAT

INPUT PROBLEMS ! ! !

Would your programs benefit from:-

- crash proof inputs
- ability to select input as words or numbers with selection prompt
- ability to predefine allowed length on input
- users defined key (activates function key on VIC/64)
- date input with automatic verification
- ability to prompt existing value of variables and alter only those characters required (if any)
- ability to layout full screen of inputs, simply and efficiently and then move up/down screen or end at any point
- audible warning of any errors
- ability to disallow any character e.g. comma

All this, and more, is possible using Compuprint's Input Routine. A professional product developed for our own business programs, the routine greatly enhances any program.

'Input' is all basic and fully adaptable. It comes on cassette/disc complete with a fully documented, line by line listing and example of use.

Available for all Commodore machines including VIC20 and C64 (also for Apple if that word is allowed in this book!).

Special Price to ICPUG Members £19.95 including VAT and P/P

Please state machine and disc format/cassette when ordering

KEEP YOUR COMPUTER TIDY WITH A

"COMPUPRINT COMPUTER STAND"

For VIC/64 etc.

High quality plastic stand holds keyboard, 2 discs or cassettes, monitor with ample room for power units and usual mass of cables.

COMPUPRINT COMPUTERS LIMITED,
4, Sands Road, Swalwell,
Newcastle NE163DJ

Telephone 091 488 8936.

See enclosed leaflet for further details.

£19.95 inc VAT + £3.00 p & p.

THE VIC COLUMN

By Mike Todd

THE ADVENTURE SWAP

At the beginning of the year, Brian Roberts of Luton suggested that, because Adventure games tend to lose their interest after being successfully completed, it would make sense to set up a swap scheme for members to swap their Adventure cartridges. Interest was sparse and so we've decided to drop the idea - so, please don't write to Brian if you've got swaps.

Maybe it would be possible to organise the idea on a regional basis where members can make personal contact. I'm sure that this could be arranged through regional organisers or at regional group meetings.

If the lack of interest is not due to the fact that people don't want to swap games, then it may be that an ad in the June edition of VIC COMPUTING (p30) has attracted people. It is under the name of MAVAC ENTERPRISES (101A, Underdale Road, Shrewsbury, Shropshire) and details are obtainable by sending an SAE to that address.

I've had no dealings with this organisation, so I would recommend an element of caution before plunging in; but, I would be interested to hear of anyone's experiences of swapping through MAVAC.

THE VIC CHARACTER SET

In the May Newsletter I prepared four tables of characters for the VIC and 64, and it seems that people have been quite impressed by them. What they may not be so impressed to know is that there is an error in the two tables of CHR$ values!

In the introduction to the tables (page 285) I state that columns 6 & 7 are actually repeats of columns 2 & 3. The truth is that the tables were originally prepared for the PET, where columns 6, 7, E & F are actually undefined. This resulted in an "image" of columns 2, 3, A & B appearing here.

What I hadn't spotted was that, on the VIC and 64, columns 2 & 3 have now been defined as a repeat of columns C & D. I don't know if this was deliberate or not, but it does make the CHR$ values closer to true ASCII code than they would otherwise have been.

For instance, in TEXT mode (table on page 289), columns 4 & 5 are lower case letters, in true ASCII they are upper case letter. Columns 6 & 7 are now upper case letters (as in columns C & D), true ASCII has lower case here.

This does at least provide some compatibility if ASCII devices are used on the RS232 interface and which would, if the PET convention was followed, have produced garbage if the characters received were lower case.

I'm sorry for the confusion and my thanks to Marcus West (brother of the now infamous Raeto West) for pointing it out.

THE SPEAKING VIC

It seems to be the trend nowadays to get computers talking, whether installed in a car or on the living room table.

Speech synthesisers were once extremely expensive and generated some very good quality speech, now the price has fallen significantly and the quality of the speech, while not as good as in the earlier days, is quite acceptable.

Derek Hoare of West Green in Crawley has recently bought the Maplin VIC-20 TALKBACK speech synthesiser and has been very pleased with it. The following notes are from him.

Unlike some speech synthesisers, the Maplin unit makes its speech from different sounds and not actual words and so it is almost unlimited in its output. Each word is made up by stringing some of the 59 different sounds (called ALLOPHONES) together with any of the 5 different length pauses.

To get a sound all you need to do is POKE 39936+sound,0 where "sound" is the number of the sound you want and, although there are only 64 "sounds" available, tonal inflection can be generated by adding 64 to raise the tone and 128 to lower it.

Although the unit requires no expansion on the VIC, a large number of DATA statements holding the different speech patterns will soon use up memory.

At £25, Derek believes the unit (in fact a kit) to be very good value and it has brought his Backgammon game to life in an amazing way and the uses for the imaginative are likely to be endless.

One problem he did have was that some games tapes tended to interfere with the unit, producing odd squeaks and buzzes but this was easily solved by unplugging the unit !

THE VIC AND THE 1541

The original VIC disk drive (the 1540) has been replaced by the 1541, although old 1540s can be upgraded by replacing the ROMs. The main reason for the change was to allow the drive to work correctly with the 64 and its normal set up is for use with the 64.

If you use an uprated 1540, or the 1541, with the VIC, it is worth a reminder that the drive needs to be switched from 64 to VIC speed and this is most easily done at he start of a program when the command channel is opened as follows:

```
OPEN 15,8,15,"UI-"
```

--oOo--

THE ADMINISTRATOR

STOP PRESS
CBM 64
DATA BASE
LAUNCH

The original transaction processing system which can learn and run repetitive functions. It allows the user to create his own records and also his own menu for job selection. Widely used by all types of business.

TYPICAL LIST OF APPLICATIONS

Appointments Planning
Bank Accounting
Bonus Schemes
Book Keeping
Bureaux de Change
Contract Costing
Cost Ledgers
Credit Control
Customer Files
Diaries
Equipment Leasing/ Rental/HP
Estate Agents
Estate Management
Expense Accounting
Invoicing
Job Costing
Mailing
Medical Records
Membership Accounting
Name & Address files

Office Administration
Parts List
Personnel Records
Petty Cash Ledger
Plant/Asset Register
Portfolio Management
Price Lists
Property Management
Purchase Ledger
Rota Planning
Route Planning
Royalty Payments
Sales/Purchase order files
Service Records
Statistics
Stock Control
Test Data Storage
Theatre Event Costing
Time Costing
Vehicle Costing

and many others

Come and meet The Administrator on Stand 21 & 22 at the 4th International Commodore Show, Cunard Hotel, Hammersmith. June 9th-11th 1983.

The Businessman's Dream

STAGE ONE SOFTWARE 300 Ashley Rd. Parkstone Poole Dorset 0202-735656

MICRONET - WHAT HAPPENED ?

By Brian Grainger

I must repeat at the outset the words on the contents page of this magazine. The views expressed in this article are my own and not necessarily those of ICPUG or the Editor. Indeed the article was written prior to seeing the July Newsletter in which the Editor may well have expressed his opinion. [No, strictly the facts - Ed.].

So what has happened to Prestel and Micronet since Stephen Rabagliatti promised us this super service (Vol.5 No.2 p.132) at about 50 pounds plus subscription. Well, not a lot as far as the PET is concerned. The question is why.

The aim of Micronet was to provide a cheap means for Micro users to get onto Prestel and provide a service to computer users, especially the provision of downloadable programs. As far as CBM machines are concerned it was clear ICPUG had a major role to play and much software from the library went on the system, and is still there. Micronet started with the BBC Micro service and to enable them to get sufficient free programs available some PET stuff was converted. This did not exactly please ICPUG. What was worse was when the PET system was announced the prices had risen to up to 100 pounds for the offer price and a ridiculous 200+ pounds price at retail.

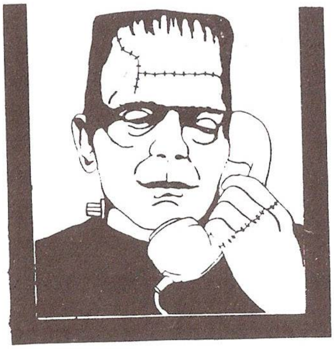
In May Micronet demonstrated their service (?) for the PET at Watford. The demo was not a total success as the acoustic modem was not working too well. A hard wired modem worked better and we got some idea of the facilities. Despite the cost and apparent doubts of the modem suitability some of us drew a deep breath and joined up. Since then there has been virtually no support for CBM users. Why ? Were Micronet not doing their job ? I thought so but remember what I said about ICPUG playing an important part in the CBM service. After all, who else has the expertise and the software to run a decent service. What was ICPUG doing ?

At the ICPUG committee meeting of 20th May the service was discussed. Virtually all the committee were against it. Reasons given were (1) too costly, (2) the modem doesn't work, (3) troubles that Ron, our Editor, had with getting his problems sorted out. My own comments that my modem worked perfectly OK, and a friends did after tuning, had little impact. I was saddened by the attitude of the committee to Micronet, but I was in the minority and there WERE some problems.

The story now moves to the ICPUG committee meeting of 13th July. When our Chairman mentioned that in discussions he had with Commodore they requested ICPUG did not support Micronet. I was amazed at this and suggested to the Chairman that ICPUG cannot go on dithering with the most exciting applications of a microcomputer now being made available. We will have already wasted 6 months by the AGM. It was agreed that the topic is to be discussed at the AGM and serious consideration be given to ICPUG supporting Micronet.

I hope this article appears before the AGM. Whichever way the decision goes at the end it will be right- provided the members of ICPUG make it and we are not inveigled into a particular course of action by Commodore. It is time to remember what the 'I' in ICPUG stands for.

--oOo--



QUOTE

You never know till you have tried.

--oOo--

CLUB SOFTWARE REVIEW - MICROMON

By Brian Grainger

I shall start this article by first of all apologising to those of you who wrote to the Librarian for copies of the programs reviewed in Vol.5 No.3 and were told they were not available. Those numerical analysis programs are now in the library on Disk 17. If any of you have trouble I always have copies of programs reviewed and a 4040 disk and return postage will always ensure a copy is on its way.

This time I wish to review software suitable for the beginners or enthusiasts in machine code. The first thing one needs when starting machine code is an Assembler and a Disassembler. The first allows you to input your code as you would write it (e.g. LDA #$01, STA $8000 ...) instead of individual bytes. The second translates bytes back into the source statements. For some time now several monitors which include these functions have been available. Probably the most well known is Supermon closely followed by the ROM based Basmon. Last year an enhanced monitor called Micromon was published in Compute. Since that time it has undergone further enhancements and is now available (Micromon and Micromon+) at revision E from the software library. As far as I am aware it is written for BASIC4 either 40-column or 80-column but I believe only minor mods are needed to convert to BASIC2.

Micromon is essentially a machine code monitor which as well as an Assembler and Disassembler allows (i) display of memory in hex and ASCII equivalents, (ii) load and save of programs, (iii) filling of memory with the same byte value, (iv) transfer of memory with or without modification of address values where necessary (which will allow relocation of machine code) and (v) display of the processor registers.

Some debugging aids are included which allow a trace of a machine code program, walking through the program step by step and setting break points. Functions exist to convert from one to any other of decimal, hex, binary, ASCII. Addition and subtraction of hex numbers can be done as well as calculating branch offsets. A block of memory can be compared with another block and the locations of unequal bytes will be identified. The checksum of a block of memory can also be found. Finally Micromon allows a search of memory for a given string of bytes or ASCII.

```
      e.g. .H 6000 6FFF 20 D2 FF
```

will search 6000-6FFF for the code JSR $FFD2. All occurences in the range will be flagged. The enhancement of version E is to allow undefined bytes in the hunt function.

```
      e.g. .H 6000 6FFF 20  ? FF
```

will search for any JSR to the FF page of memory. Any number of undefined bytes can be specified.

With many functions pressing the cursor keys continuously will cause the listing produced to continue forwards or backwards. This is particularly useful for scrolling (in both directions) of memory display or disassembled code. However it also works for say hex to decimal conversion, cursor down causing the hex number converted to increase by 1.

Micromon plus adds to Micromon facilities for controlling peripherals. DOS support is available from the monitor as well as load and save commands specifically for disks. Disassemblies or memory displays can be sent directly to a printer as well as controls for turning on printer paging, with or without headings. A simple .P command exists to switch output from screen to printer and vice versa. Any other device can be specified with say .P 06 for device #6. Control characters can be sent directly to the output device, so that for example .↑S will clear the screen.

The load address of a disk file can be found by the to load a file starting at an address specified by the user rather than the program. Very useful for breaking protection on programs which load onto the processor stack!

As you can see this monitor is very powerful for the machine code enthusiast. Because it lacks BASIC programming aids it does not make BASMON and PLUSDOS obsolete but it goes a long way to providing some of their facilities. The club also has the original source code for Micromon should anybody wish it. It is written in Moser Assembler format at present but I hope that it will be converted to Commodore Assembler format in due course by Joe Griffin. He spent many hours doing this to version C only to find on completion that version E existed! The user group responsible for Micromon have an interesting philosophy. Group members get the latest versions. Non-Group members who ask for copies get them but two revisions out of date !

One final point for enthusiasts. I have long been using an early instruction manual for the Commodore Assembler which did not identify conditional assembly instructions available in the latest versions. These are the .IFE and .IFN directives. The former tests for an expression equal to 0, the latter not equal to 0. An example follows:

```
BASIC=4 ; set up BASIC ROM no.
.IFE BASIC-4 <
STALK=$F0D2 ; send talk for BASIC4
>
.IFE BASIC-2 <
STALK=$F0B6 ; send talk for BASIC2
>
```

By use of the above one simple change of BASIC to 2 will allow the code to be assembled for BASIC2. A very useful idea which is used throughout the Micromon source.

--oOo--

THOUGHT FOR THE MONTH

You will never find time for anything. If you want time, you must make it.

Charles Buxton.

--oOo--

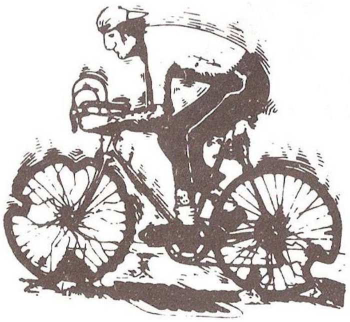
IEEE-728

Some of you may be familiar with the IEEE-488 standard, if only because Commodore use a near implementation on their up-market computers. Basically the standard defined the interface in terms of the mechanics of the hardware and the handshaking of data across the interface. The standard makes no recommendations regarding the content or format of the data. This is now covered by a new standard, IEEE-728. Users who design, assemble or program IEEE-488 systems will find this document invaluable in dealing with devices which generate, process or interpret data by a variety of codes and formats.

Copies of this important new standard are now available from the WASEC GPIB Accessory Service. Contact the Sales Office, WASEC, P.O. Box 161, Wallington, Surrey, SM6 8BA. Tel: 01-647 3199.

R.D.G.

--oOo--



so long!!!

Printed and distributed by COMPUPRINT COMPUTERS LTD.
4 Sands Road, Swalwell, Tyne and Wear. Tel (091) 488 8936