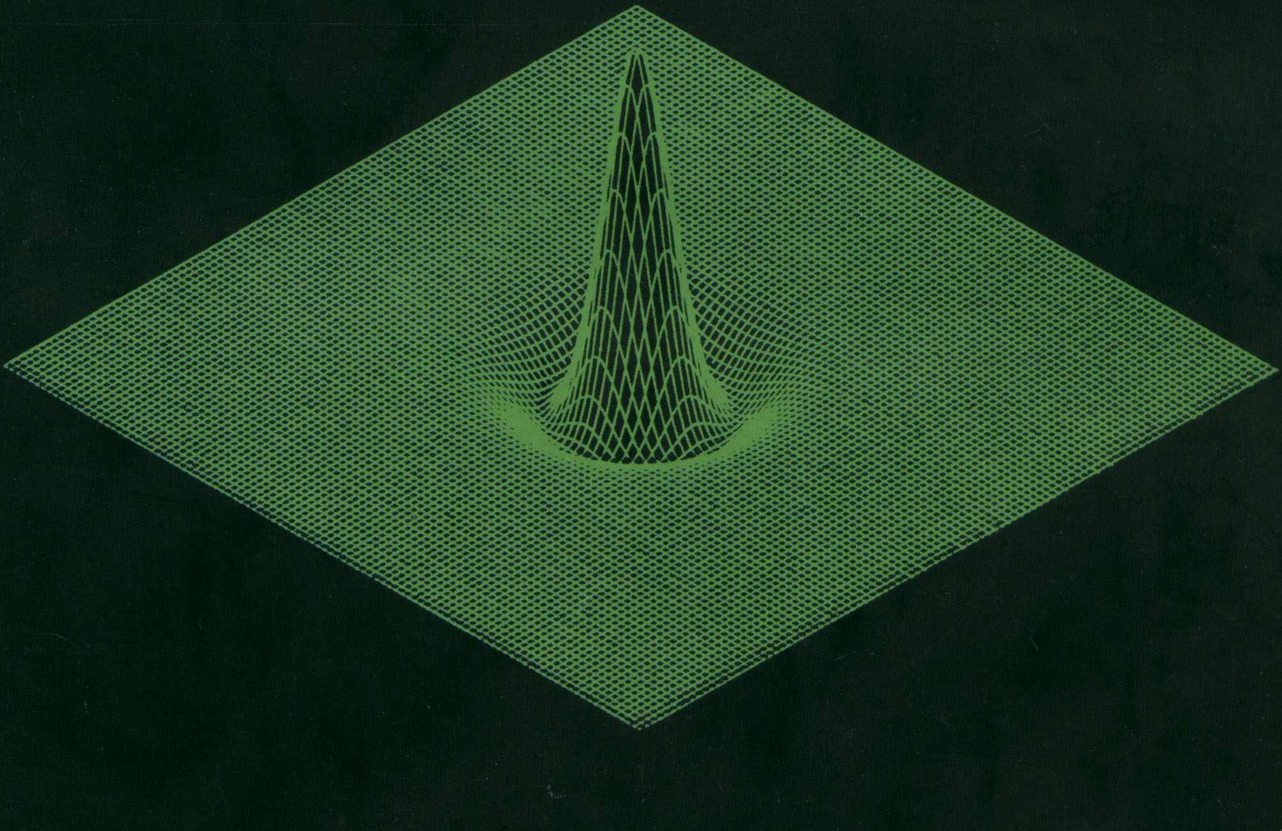
commodore

COMPUTING

international

October 1982 £1.00



The independent magazine for Commodore computer users

damsdamsdamsdams
damsdamsdamsdams
Business Computers Ltd.
Tel: 051-548 7111 20 lines

VICAID

Two sets of Fabulous Utilities in one!

PROGRAMMERS TOOLKIT
Gives extra commands: Auto, Number, Help, Delete, Change, Trace, Step, Light Pen, Break etc.

and

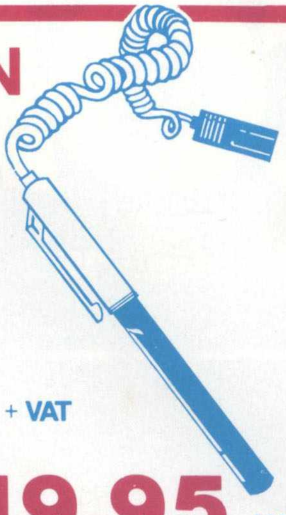
MACHINE CODE MONITOR
Gives Save, Memory Display, Load, Verify etc.
Similar to TIM on PET.

Examine the VICS ROM

Needs DAMS RAM/ROM board or similar

£19.95 + VAT

LIGHT PEN



VIC
LIGHT PEN

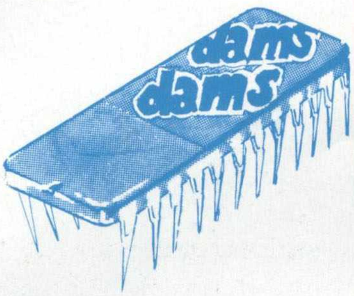
**DAMS PRICE
ONLY**

£17.35 + VAT

FOR PET
12" SCREEN **£19.95** + VAT

VICMON

**THE ULTIMATE
PROGRAMING AID
FOR THE VIC**

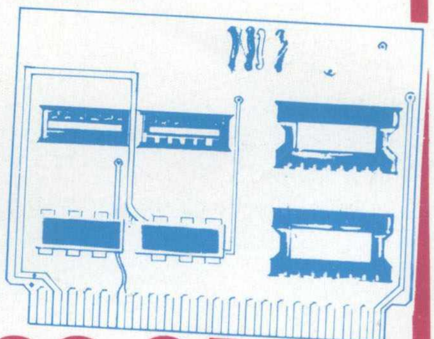


Full machine code package with:
Assembler, Dissassembler, Fill, Re-locate, Identify, Exchange, Compare, Printing, Dissassembler etc., etc.

Needs DAMS RAM/ROM board or similar

£19.95 + VAT

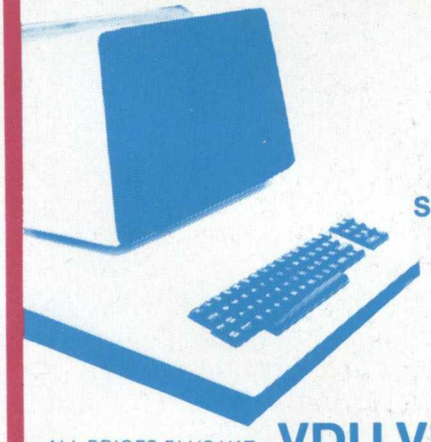
RAM 'N ROM BOARD



3K RAM
In Hires area.
Also space for VICAID and VICMON programming aids.

£22.95 +VAT
(Includes Cover)

**BUY THE 3K RAM 'N ROM BOARD WITH VICAID AND VICMON WITH MACHINE CODE MANUAL (WORTH £5.00) FROM MOS TECHNOLOGY FOR ONLY £67.85 + VAT AND GET A FREE VIC LIGHT PEN (WORTH £17.35)
VIC REFERENCE GUIDE R.R.P. £14.95 DAMS PRICE £14.50**



VIEW THRU

**ANTIGLARE
SCREENS FOR PET**

40 Column
£17.95
80 Column
£19.95

ALL PRICES PLUS VAT

VDU VIEW THRU

VIC STARTER KIT

VIC 20
C2N Cassette Deck, 10 Blank Cassettes, User Manual, Vic Programmers Reference Guide, 1 Joystick.
Worth £238.30 ONLY £214.00 +VAT
(VAT INCL. PRICE = £244.62)

OR VIC 20
With free 3K RAM pack or Super Cartridge Game **ONLY £173.83** + VAT

VISA

ACCESS &
BARCLAYCARD
WELCOME

DAMS BUSINESS COMPUTERS LTD.
Gores Road, Kirkby Industrial Estate, Kirkby, Liverpool L33 7UA.
Telephone: 051-548 7111 (20 lines)

Contents

Editorial

Editor
Pete Gerrard

Advertising Manager
Peter Chandler: tel 01-439 3537

Production
Lithotype Design
London EC1.

Managing Editor
Nick Hampshire

Commodore Computing is published 10 times per year by Nick Hampshire Publications. It is not in any way connected with Commodore Business Machines U.K. Ltd.

Typesetting by
Centrepoint Typesetters Ltd, London
Printed by Spottiswoode Ballantyne Ltd, Colchester and London.

If you would like to contribute to Commodore Computing, please send articles or programs to:—

Commodore Computing International
193 Wardour Street,
London W1

We will pay 10 pounds for each program printed, and 20 pounds for each article published, which should be approximately 1,000 words long.

To many people the Pet is a computer capable performing just one task. In our own office, someone who'd never seen the Pet being used for anything other than wordprocessing was convinced that that was all it could do, and it took a lot of persuasion to make her realise that the wordprocessing program was just that and that alone: a program to enable the Pet to emulate a wordprocessor.

After a demonstration of a data base program and various other packages it was fairly clear that the Pet is a computer capable of many, varied jobs. A look through Commodore's own Approved Products catalogue would show just how many different uses a Pet can be out to: from accountancy to typesetting, from statistical analysis to space invaders, and frontiers beyond.

We hope that this diversity of interests is reflected in the contents of the magazine. By the very nature of the magazine we have to cover a lot of varied topics: what applies to the V10 will not (usually) apply to the SuperPet, for instance.

Certainly that diversity is shown by a look at a cross-section of you, the readers. From private individuals to the largest company conglomerate, there must be an awful lot of Pets out there doing some very strange things!

This, in a computer industry that is constantly changing must be a good thing. The occasional manufacturer who places all his eggs in one basket (e.g. Clive Sinclair, one ZX machine at a time), will succeed, but these are few and far between.

By taking advantage of the wide range of interests shown by the market place, and producing a computer to fill every available gap, a company will naturally be in a stronger position to survive. Commodore are attempting to do this with a variety of machines, from the Vic 10 (or whatever it will become known as) to the 720 series.

As Commodore continues to grow, so must we, the magazine. Whatever system you own or use, we look forward to having you with us!

Letters

Dear Sirs,

In the July edition of Commodore Computing you published a routine for disc appending files, written for Basic 2 machines, and asked for a Basic 4 version. This routine obviously fulfills a great need, and as my system is Basic 4 I undertook a conversion of Mr. McLean's program. Much to my relief, and surprise, it proved a simple operation, and the machine code and a basic loader (for those wary of the monitor!) are attached.

The code is fully relocatable to any suitable memory area. In the version below I have placed it in the first cassette buffer, which is convenient for my system.

I have a question that you, or your readers, may be able to answer. It seems that the 12'' screen CBMs have a CRT controller chip which potentially has facilities for the connection of a light pen, although this is not implemented in the CBM. Does anyone know whether this facility can be used, by hardware modification of a relatively simple sort, and if so, how?

Yours sincerely
Dr. Heathcote
Cardiff

Dear Dr. Heathcote,

Thank you for the listings, which we reporudce below.

The chip in question is the Motorola 6845, which does indeed have the facility to connect a light pen, but this is not implemented on the Pet, and as far as we know no-one has even attempted to do so. This chip is in fact implemented on the Pet, and as far as we know no-one has even attempted to do so. This chip is in fact pretty impressive, and it's a shame that Commodore didn't make full use of the facilities on board.

To go into further detail about the 6845 would probably fill the magazine, and for many months to come at that, but additional information can be gleaned from a splendid book called Microcomputer Components, published by Motorola themselves. You should be able to get a copy from any distributor of theirs, and inside you will find some 20 or so pages all about the 6845. An amazing bit of hardware!

Dear Sirs,

Please find a cheque for one years subscription to Commodore Computing.

I am an avid Vic fan, and a proud owner of such a machine. Could you tell me if the Vic 20 can be simply (and hopefully relatively cheaply) upgraded to such machines as the Commodore 64 when it appears, and will your magazine be covering Commodore's new machines as I have heard many rumours (nice ones at that) about the above named micro.

Yours sincerely
G. Hughes
Blackpool

Dear Mr. Hughes,

Thanks for the letter, and subscription cheque. We hope you enjoy the magazine!

We certainly shall be covering any new machines that Commodore produce. We are called COMMODORE Computing after all! Seriously, our aim is to expand the magazine as the range of machines available expands, so as to give the best coverage we possible can to each and every one of them.

The question of upgrading existing VIC 20s is an interesting one. Commodore's John Baxter has said that an upgrade will be available, and if he gets his way it will cost around 100.00 pounds (remarkable really: the VIC is just under 200 and the 64 is just under 300!). However, the Vic has outlet for one joystick, the 64 has two. There are other differences as well, about the only common thing being the keyboard.

So whether this upgrade means ripping out everything from the Vic 20 and being a total replacement (keeping the keyboard of course!), or what, we don't yet know. We'll keep you posted.

Dear Sirs,

I have a CBM 3032 which was acquired to help a schoolboy with computer studies. Now he is trying to get hold of software and program books to play games on the machine, as well as learn to program and operate it himself (dad's interested as well!)

Any advice, sources, books you can recommend would be appreciated. I often wonder whether tried and tested games programs have every been collected into one book and published for the 3032. Have they?

Yours sincerely
T.J. Davies
Redruth

Dear Mr. Davies

Thank you for your letter.

Many people decry games as being of no use to man nor beast, but as an initial introduction to computing I think that they serve a most useful purpose. It is no coincidence that Commodore's (and other people's) best selling educational tapes, for instance, have had some kind of game-playing role attached to them.

Consequently, although there are many tapes available, from people like Audiogenic (tel. 0734 586334) or Supersoft (tel. 01-861 1166), it comes as some surprise to find that there are remarkably few books of games listings around. Certainly for the 3000 series machines anyway.

Still, there is one, called Pet Fun and Games, published by McGraw/Hill. It is aimed at the Commodore range generally, rather than just the 3000, but the programs will nonetheless work on all machines. Well put together and well presented, a telephone call to 0628-23431 will give you all the information you need.

```
027A A9 00          LDA £$00
027C 85 9D          STA $9D
027E 20 7D F4       JSR $F47D
0281 A9 60          LDA £$60
0283 85 D3          STA $D3
0285 A4 D1          LDY $D1
0287 D0 03          BNE $028C
0289 4C 00 BF       JMP $BF00
028C 20 49 F4       JSR $F449
028F 20 A5 F4       JSR $F4A5
0292 20 D2 F0       JSR $F0D2
0295 A5 D3          LDA $D3
0297 20 43 F1       JSR $F143
029A 20 C0 F1       JSR $F1C0
029D 20 C0 F1       JSR $F1C0
02A0 38             SEC
02A1 A5 2A          LDA $2A
02A3 E9 02          SBC £$02
02A5 85 FB          STA $FB
02A7 A5 2B          LDA $2B
02A9 E9 00          SBC £$00
02AB 85 FC          STA $FC
02AD 20 8C F3       JSR $F38C
02B0 4C 1C F4       JMP $F41C
```

```
 10 PRINT" BASIC 4 DISC APPEND"
 20 PRINT"THIS ROUTINE WILL ALLOW A SUBROUTINE"
 30 PRINT"SAVED AS A PROGRAM FILE ON DISC TO BE"
 40 PRINT"APPENDED TO A PROGRAM IN THE MEMORY"
 50 PRINT"N.B. THE SUBROUTINE MUST BEGIN WITH"
 60 PRINT"A LINE NUMBER GREATER THAN THE LAST"
 70 PRINT"IN THE MAIN PROGRAM                  "
 80 PRINT"ACTIVATE WITH :-
 90 READI
 100 PRINT"SYS"I;CHR$(34)"FILE NAME"CHR$(34)",8"
 110 FORJ=0TO56:READ A:POKEI+J,A:NEXT
 120 REM THE ROUTINE IS FULLY RELOCATABLE
 130 REM THE START ADRESS IS IN LINE 1000
 1000 DATA 634
 1010 DATA 169,0,133,157,32,125,244
 1020 DATA 169,96,133,211,164,209,208
 1030 DATA 3,76,0,191,32,73,244
 1040 DATA 32,165,244,32,210,240,165
 1050 DATA 211,32,67,241,32,192,241
 1060 DATA 32,192,241,56,165,42,233
 1070 DATA 2,133,251,165,43,233,0
 1080 DATA 133,252,32,140,243,76,28
 1090 DATA 244
READY.
```

New Product News

LATEST Commodore News

Despite the oncoming launch of the new machines from Commodore, peripherals and add-ons for the existing Vic 20 continue to appear with great regularity.

D.A.M.S. up in Liverpool (051 548 7111) have brought another light pen onto the scene. Admittedly at 19.99 pounds it is cheaper than the rest (probably because it uses less cable than any of the others, albeit sufficient), but apart from that what else can you say about a light pen? It works, there is a demonstration program supplied (which works in interesting ways its wonders to perform), and that is about it.

Still, it is on the market, at a fairly cheap price, so prospective buyers of light pens could do worse than examine this one.

One would never know that Audiogenic (0734 586334) had a new person handling their public relations. Bombarded with press releases, they do bring to light a number of interesting products.

They've just taken on board the Supersoft product Mikro Assembler, suitably modified for the Vic. Some of the facilities provided include an internal monitor and an assembler, to make machine code programming on the Vic significantly easier. As well as this, there is a high resolution plotting facility (based on a 176 by 160 matrix), and additional control over the Vics existing sound capabilities. Available on a cartridge, with an extra 3K RAM, for just 49.50 pounds.

Next off the line comes something called BUTI, which they insist you pronounce as Beauty: this is what we in the trade call a chip Buti. In their now traditional format of coming in the form of a plus in cartridge with an extra 3K RAM, Buti retails at 39.99 pounds. Essentially a variant on the old Toolkit idea, this adds an additional 17 commands to the Vic's existing Basic.

Moving smartly on, we come to Boss, described as the definative chess game for the Vic (but then aren't they all? When will Boss play Commodore's own version of Sargon?). Features implemented include castling, 'en passant' and queening (but we chess players know that you don't always want a queen!), at 14.99 pounds it does seem to be a fair buy. In a competition between itself, Sargon on the Apple, PetChess on the 8032 and Chessmaster on the TI 99/4A it came out well on top. We'll try and feature a full review of this in the coming months.

Before we turn this into the Audiogenic fan club, we'll finish by mentioning Wordcraft 20 for the Vic, again as a cartridge, but with a whole 8K extra this time.

The press release starts off somewhat grandly by proclaiming that 'the typewriter has now become completely obsolete'. Well. There is no doubt that Wordcraft 20 is a well-written and well-documented package, with most of the features of its big brother Wordcraft 80. However, the merits of owning a wordprocessing system that allows you only to see 24 characters in a line at any one time (I know it scrolls up, down, left and right) I will leave up to you.

If you want to do wordprocessing on a Vic Though, for 125 pounds this is certainly worth examining.

VATman and Robbins

One of the major problems facing any retailers using the familiar Simplex cash book is ... using the Simplex Cash Book! Although the system has been running now for something in the region of

New Product News

fifty years, it is still a time consuming process to actually fill the book out, and many errors will tend to creep in.

There is also the perennial problem of the VAT returns, a harrowing financial nightmare for any accountant, however well trained.

Now however, there is a package called Micro-Simplex, which as you might surmise is based on the Simplex book. Thus, anyone who's used to using the book should take to the microcomputer equivalent very quickly. Designed by Catlands Information Systems in Macclesfield, the package is now being handled by Micro-Simplex, themselves an offshoot of Catlands. Mike Dawson is the man in the know about this, and he can be reached on 0625 61500.

Essentially then this is of interest to the retail market, and we'll be bringing you a full review of this shortly, hopefully in next months issue.

Faulty Towers?

For a long time now there has been just one major suite of programs on the Pet for hoteliers, namely the series of programs from Landsoft.

Recently we received news of another system, this one being from Devon Computers (0803 526303). Everything in the promotional sales literature points to it being along roughly the same lines (just what is a hotel package supposed to do after all? Manage a hotel), so comparison between the two at your local dealers would probably be worthwhile if you're interested in this kind of thing.

However, when sales literature includes phrases like 'finest system available in the Commodore range for Hotels', 'review this fine product', 'put you in touch with some of our satisfied users', I for one tend to fall asle ..... !

More New Chips

Kevin Viney of ICPUG has developed a trio of chips for the Commodore range of machines, and the latest of these (known appropriately as TRIOS) is now being actively marketed by a company called Consup (tel. 01-670 4411).

As is now standard with these chips, they replace one of the existing Pet ones, thus not losing any valuable space in the computer. This particular one goes into the 'E' ROM slot, and as well as retaining the Pet Basic adds a number of unique ones of its own.

For instance, the ability to scroll backwards through a listing, and to page through in either direction (a most useful routine), is presented here. Unfortunately that is basically it! One or two other things, such as shifted return deletes everything on that line after the cursor position (but does not erase it from memory until return itself is pressed), OLD is a useful command if you've happened to type NEW by mistake, and DO ignores a line number and a REM, and executes any Basic statement following.

So, for 27.00 pounds, backward scrolling of a listing could be yours!

Training Courses

Yet more training courses have been announced recently for the Commodore range of micros, this time by a company called MicroTraining, based in the West Midlands – tel. 021-550 1827 for further details. These training sessions are usually carried out in two five hour blocks on site, or you can opt for attendance at the MicroTraining computer school in Ward End, Halesowen, for five, weekly, two hour sessions.

The aim of these courses is to introduce the businessman to the use of computers, and acquaint him with just some of the workload that microcomputers can take in the office environment.

Interestingly enough, MicroTraining claim that 'Commodore were so impressed by the courses that they appointed MicroTraining to be the only external authorised Commodore training centre'. I wonder what McDowell Knaggs and Associates in Worcestershire would have to say about that?!

Computer Aided Design

Jentech Services Ltd. (tel. 07462 5287), have recently brought out a complete system for microcomputer aid draughting, and for around 5,800 pounds you get a 32K Pet, a minimum of 1/2 megabyte of storage on disk, a great deal of sophisticated software, and an intelligent X-Y plotter.

As an inexpensive approach to C.A.D. this system offers more than most, and at a very reasonable price. Upward expansion from the basic set-up is available should it be required. You can build up a library of drawings, shapes and standard parts or symbols, to be called up as required, stores on disk, scaled up and down: in other words, to be done with as you wish! A comprehensive package.

Words on Word Processing

Last month we mentioned Superscript from the Independent Commodore Products Users Group, but also said that there was some confusion over the future of the program, what with the odd court case pending. Superscript has now been taken over by a company named Precision Software (01-330 7166), and is being actively marketed by them.

It's interesting to note that Commodore thought enough of Superscrpt (including praise from Jack Tramiel himself!), that they asked the author, Simon Tranmer, to produce a version for the Commodore 64, so expect that out before long.

One bone to pick with Precision Software though: when ICPUG had the program it cost 35 pounds to buy, but Precision were talking about a price in the region of 240 pounds, which defeats the object somewhat.

Hopefully that price will come down: ICPUG started a nice precedent by releasing a really powerful, low cost word processing package, and not it's been taken out of their hands.

Whilst we're talking about word processors, another package has appeared from Landsoft (01-878 7044/7), in the form of a new version of Wordform (II this time).

Based on their earlier Wordform, it is very much a 'What you see is what you get' word processor, using as it does the screen as a typewriter: no bad idea, when introducing secretaries unfamiliar with computers to the are to using one. In other words, as you type a screen 'window' moves its way along in front of you, performing a carriage return whenever you get to the end of a line, taking any word with it as necessary.

An easy to understand package, and at 150.00 pounds probably the cheapest major program now on the market, with Precision Software having altered the price of Superscript. Well worth a look at.

Diplomacy

When you think of a really nice name for a new Pet product, perfect it, do all the literature, and get prepared for the big launch, it must be a major disappointment when you realise that the name is not, after all, quite right.

Such is the case with the Versatile Interface Card, now being distributed in the U.K. by Corrtronic, in Manchester (tel. 061-865 5070). A nifty little name you might think but what are the initials? V.I.C., the name of a certain well-known other computer from Commodore, unfortunately. So, the VIC-2 has now become known at the Diplomat.

This is a card which fits inside the existing Pet case (although optionally outside), and provides many additional external interface capabilities. On board we have 80 I/O lines (the equivalent of 8 Pet User Ports!), 2/16 bit timer counters, and 2 serial I/O ports.

In addition to all this we also set a 1K battery backup RAM, a built in calendar/clock, a power down detector, and one or two other things as well.

At a cost of 380 pounds this is worth a look if you're into interfacing to the outside world.

Program Storage

As an alternative to the usual disks or cassettes, a new form of program storage has arrived from Microscience in Stockport (tel. 061-477 3888). As is the case with Diplomat, this takes the form of a board that fits inside the Pet, and has the ability to store from 2K to 28K, using a range of 2716 or 2732 Eproms as necessary, depending on the size of the program to be stored.

As an additional security there is the facility to have your programs (which can be written in Basic, machine code, compiled, or a combination of all three!), once stored on Eprom, run automatically on computer switch on. Alternatively, programs can be called up with our old friend the SYS command.

New Printers

Ever since Commodore disbanded the old 8026 and 8027 daisy wheel printers, the market has been waiting for a replacement to come along.

None such has appeared from Commodore, but two new printers have come out recently, and the first of these bears a remarkable resemblance to the 8026: hardly surprising really since it's virtually the same machine! However, there are one or two nice additions which our old friends never had.

We now have 10, 12 or 15 characters per inch, and the ability to accept paper up to 17 inches wide, with a 14 inch print width. Bi-directional printing (at last!) gives us a much faster print speed, and the rather nice 4K buffer allows the unit to happily print away in its own time, while you get on with the host computer.

It comes with either IEEE, RS232 or Centronics interfaces, so it should link to just about anything, and if you're after further information on this 1098 pound printer ring Dataplus Ltd., on 0242 30030/37373, and say 'what can you tell me about the Scripta II?'

How many daisy wheel printers have you seen recently, priced at 485 pounds? Well, Butel-Comco have brought one such into the world, having had a few words in the ears of Smith-Corona Typewriters. Linking up to the old Pet quite easily, it is a true daisy wheel, with two versions currently available: 10 characters per inch or 12 characters per inch. You must specify the one you want at time of purchase.

An admittedly low print speed of just 120 words per minute (or about 10 characters per second) is not one of its strongest features, but at 485 pounds one can't really complain. We'll bring you more on this one next month, but meanwhile ring 01-202 2277 and ask for Cliff Osborne.

Club News

Two User Groups

There are many independent user groups up and down the country. Their enthusiastic members meet regularly once a month (well, fairly regularly), to discuss and demonstrate the Commodore range of machines. These meetings are a great opportunity for Commodore users in a particular area to get together and pool their information.

Nobody knows everything about computing, so these get-togethers give experienced users, and perhaps more importantly the not-so-experienced users, the chance to swop ideas, listen to guest speakers, and watch demonstrations of both hardware and software. All the latest news from Commodore, and the industry in general, are poured over at these gatherings, which, apart from the computer aspect, make a very pleasant social event.

This month we take a look at two of the independent user groups in the south of England, namely the Canterbury group and the Watford group.

Canterbury Group

The Canterbury group meets on the first Tuesday of each month, at the Physics department of the University of Kent. It began its life back in the summer of 1980, and was officially affiliated to I.C.U.G. in July1981. With a starting list of just five members, this has now risen to forty and, as they say, is rising every month.

A typical club night involves a broadcast of the latest news in the industry, and also includes news of the South East group (featured a while ago on these pages), with which there are close ties.

After a break for refreshments, they usually feature a guest speaker giving a talk or demonstration of some new item of hardware or software : a recent example was a showing of The Administrator, by Stage One Computers. Other guests have included the knowledgeable Harry Broomhall, and the General Secretary of I.C.P.U.G. Jim Tierney.

The evening is finished off with a question and answer session on whatever topics are brought out into the open. Usually they'll round off at about eleven o' clock, having spent some three and a half hours covering all the bases.

Members are kept well informed of forthcoming meetings and events by a monthly circular. As with all these user groups, their members come from many walks of life: Canterbury in particular features children, school teachers, businessmen, and many more. In short, everyone is welcome.

If you live in the area, own or use a Pet or Vic machine, and would like further details, your person to contact is John Bickerstaff, at 48 Martin Down Road, Whitstable, Kent. He can also be reached by telephone on 0227 272702 at home, or during the day on 01-499 9102.

On to Watford

Another well organised club is the Watford group, run by Stephen Rabagliati and Rod Eva. After a recent telephone conversation with Stephen we were given a rundown on the history of the club, and how it has progressed to its current form.

The club is now a year old, and was started just after the Pet Show of 1981. Stephen and Rod decided that they would like to form a group of their own, and so with help and ideas from Mick (of the South East regional group), the club began its life.

The existence of this new club was soon on the streets via the newsletter Oz (no, not that one!). Other promotions included a mailshot to all known owners and users in the area, asking them to come down to the meetings.

The first meeting had an encouraging start, with forty people turning up. This continued for the next three months, but each month the forty people were different . . . a worrying time for Stephen and Rod. Everything finally settled down, and now the club has seventy paid up members (membership being just three pounds per year).

The club has five Pet machines, allowing members lots of hands-on time, obviously a great benefit. The majority of members are businessmen, so naturally the emphasis is on business hardware and software. Recent work has involved the use of Prestel on the Pet, and other projects have included help in preparation of the I.C.P.U.G. software library. Obviously an active and enthusiastic bunch, they also involve themselves in exhibition work, both locally and nationally.

Meetings are usually on the second Monday of every month, and in addition a monthly newssheet is sent out informing members of changes, general information and personal requests for help.

Future plans include a separate group for the Vic machine, so if you live in the area, own a Pet or a Vic machine, and want to get involved, please contact Stephen, care of Grocery Distribution, Grange Lane, Letchmoe Heath, Watford.

If you would like your club featured in a future issue of Commodore Computing International, please get in touch with the editor of the magazine at the address on the masthead.

Education

Computer Programming History

I recently read an account of the sinking of the Bismarck and then began to realise how this age of computers can subscribe towards today's presentation of history and also to posterity.

The Bismarck epic is typical of countless others on land, sea, and in the air inasmuch that a great deal of documented orders, times, positions etc. is available. It is from such detail that computers can be programmed to virtually re-create historical events such as this, and here therefore exists a means to present and preserve history with proved accuracy because for example, if orders for speeds and headings have been correctly recorded, a computer will produce tracks, courses and engagement areas to scale. They therefore are also capable of producing a 'living' diagram of land, sea, and air battles and this is particularly significant with respect to visual displays of history which may provide both interesting television material and accurate reference for future historian.s.

Take the Bismarck action as an example. A computer programme based upon documented times available enables this epic to be presented literally to any time scale. This means that the five days elapsing from the time fo the Bismarck's sailing from Bergen until its sinking at 10.40am on May 27th 1941 can be scaled down to a (say) one-hour programme. Thus, a coloured 'live' diagram graphically displaying distribution of forces involved and their courses of action serves as a fundamental reference throughout a television programme of the sinking of the Bismarck.

It is at this point an ability to scale time adds further facilities for clarifying this major event in World War Two. For example, while a one-hour programme must display progress or events at a rate 120 times faster than the actual time ensuing between the 22nd and 27th May, this is a practical rate for displaying an overall reference diagram of television screen size. On a 20-inch screen for example, the distance covered by the Bismarck from Bergen to its final resting place in the Atlantic is approximately 21 inches i.e., its position on a television screen changes at an approximate rate equivalent to the tip of the minute hand of clock, seven inches in diameter. While it is to be appreciated that five or six other deplying forces are to be simultaneously displayed and indicated during a commentary. At certain stages also, the programme will be re-scaled in time to enable detailed accounts of major incidents to be illustrated.

The fundamental programme in addition to providing a realistic continuity, also serves to trigger existing historical pictures and films into the television programme at appropriate times including their actual times of happening relative to the basic time scale. Furthermore, and in order to augment this type of presentation, computer techniques today enable events to be graphically re-constructed. For example, a computer programme may be re-scaled in time and in graphics to produce separate diagrams where warship outlines, ranges, and shell trajectories are illustrated along with vulnerable target areas, hits, torpedo strikes and their subsequent effects. At this stage however, a programme should not be just a technical exercise but a vehicle for respecting and remembering thousands of dedicated and brave souls who were lost at this time.

It is not impractical to forecast that the day will come when much of history will be re-created using computer techniques including three dimensional displays of battle areas on land sea and in the air. Here is a beginning using two dimensions and if to some, a re-creation of the past is wasted time and energy, perhaps it is as well to remember it is the past which determines our future and this should therefore be clearly and accurately recorded.

Microcomputers in Business·

Taking the Next Step

Last month we presented a few introductory tips on what to look for, and indeed what to look out for, when going about purchasing for the first time a microcomputer system for use in the business market.

This month we'll go one step further, by assuming you have your system installed and running in your office, and that it's quite happily solving the major problem you bought it for. However, in the back of your mind is the nagging feeling that this is, after all, a microcomputer, and shouldn't it be doing something else as well?

But all you know is how to press SHIFT/RUN-STOP to set the program running. How do you go about taking the next step? That's the problem we'll try and solve.

Training Courses

We have reported in past issues of Commodore Computing on the large number of companies that are now running training courses based on Commodore equipment. You only have to look at virtually any New Product News section to see just how many there are.

This is probably one of the best first steps you could take. These courses can last for any length of time from one day upwards, although two or three is the more usual schedule. Cost will obviously vary from place to place, so it's worth having a look around to get the best value for money.

The sort of material covered will again depend on the course, but as an initial foray your best bet would be to try one that was a more general introduction, rather than an attempt to teach you Basic programming in three hours (they exist!).

To begin with, you won't really want to get into programming. First of all, you'll need to know whether it would be worth your while to do this, so get an ideal of what your micro is capable of doing from the people who know, in preference to deciding yourself what you think it's capable of doing.

Next Steps

Having obtained at least some idea of what the Pet can achieve for you, one then arrives at the question: how do I go about implementing it?

There are three major options available to you, and the final decision must depend on cost, time and your own personal aptitude.

The first option would be to do it yourself. Certainly cost would be a minimal factor here, since you're hardly going to bill yourself for programming work done on behalf of your own company (are you?!). A look at this month's Applications Story will give you an idea of how someone else has tackled this quite difficult task, particularly if you've never programmed before.

If this is the course you decide to follow, then you're going to have to learn about programming for yourself. As stated earlier, there are a number of courses open to you, any number of which can take you into the mysterious depths of programming, both in Basic and machine code. Whether you want to go on any of these will be entirely up to you. Out of the three choices I personally would go for this one. Firstly, you will be able to implement not only immediate changes, but any future ones, yourself. Secondly, if at any stage you decide to install more equipment, you will stand less chance of being bamboozled by the slick salesman. Finally, it will not cost you the sometimes quite large sums of money that our second option will cost, and this is ...

Microcomputers in Business·

Outdside Help

There are many programmers, software houses, dealers etc. who would be only too willing to customise existing and future programs to your specific requirements, so that you could have no quarrels over the functions those programs would perform. You stated what you wanted the program to do, and they made it do the job for you.

Sounds ideal, doesn't it? There are, needless to say, pitfalls.

Principal amongst these is the price. Custom programs do not come cheaply, and even by doing a lot of shopping around you are not going to reduce the price by much. Essentially, everyone knows what everyone else is charging, so its up to how competent you think the programmer is rather than by how much you think he's trying to line his wallet which should be your ultimate guideline.

You will pay a lot of money, but on the other hand you will get a program that does its job satisfactorily. No more, no less : you will only get what you ask for.

In many ways this is not an ideal solution: it's expensive, is the main downfall, and you could really do it yourself if necessary. Still, not everyone has the time to do it themselves, so the decision is yours.

Off the Shelf?

And so onto our third option, namely buying a package that is available off the shelf.

Software packages exist to cover just about any commonly encountered computing requirement, and even most of those which are extremely uncommon! Certainly, the majors of word processing, stock control, accountancy etc. are all admirably covered by a host of companies. The difficulty begins when you try and settle the one that's right for you.

Presumably, you'll have started with such a package anyway, so an important area to look at will be the interfacing from one package to another. For instance, just about anyone who produces a generalised data base package has realised the importance of interfacing that package to a word processor.

Most of the accountancy packages around have a similar link. The well known Visicalc program will interrelate to just about everything currently on the market. This could save you an awful lot of time when working on generalised information.

As a final example, Microfacts82, reviewed last month, is a totally integrated accountancy package that has links to a stock control, and many other options as well.

'Our Computer Is on the Blink. Can You Send Over a Hundred of Your Fastest Mathematicians?'

So, always be conscious of this facility: future expansion could be severely hindered if not enough thought is given to this in the early stages of office development.

Other Considerations

There is a great temptation to purchase a package simply because it is there, rather than because you think it will benefit your office procedures. Rather, it is best to go for a modulated approach that fits in with what you already have, that can fit in with the existing software, and that is not going to involve starting off a whole new system the minute it comes into the office.

Whichever step you take is, of course, up to you. Rest assured that, wherever you start, it will not end there!

This is why it is important, as we mentioned last month, to cultivate a good relationship with your dealer. A helpful dealer isworth his weight in gold, and can be an extremely useful man to talk with whenever you're considering doing anything to your system, however trivial it may seem to be to you.

Obviously, dealers are in for the money: quite simply, they have to be. We've all got to make a living somehow! But, if you establish that vital rapport with him, you're far more likely not to be charged the earth for any particular job that required doing. He knows, as well as you do, that given good service you'll be back again.

Conclusion

Buying a system (hardware and software), and especially for the first time, is not an easy decision to make. Expanding that system is probably even more of a daunting task.

It is no use plunging into the dark unassisted. Read all you can beforehand, and having acquired your original system cultivate that dealer relationship: you'll find it very useful.

Software Review

DMS Diamond

An estimated 20% of all Commodore PETs in commercial environments have a DMS package, which probably makes DMS the top selling program in the U.K. Over 3,000 people are using it already, to handle some of the most diverse computer applications ever encountered. Some of the more standard uses include personel records, stock, library, student, policy, property, client, vehicle, medical and laboratory records, job costings and mailshots. Much more interesting to find are parish records stored on a computer, a refuse collection rota schedule, video library lists, and even a parrot breeders stock list!

DMS stands for Data Management System, and any type of data may be stored on DMS. Once the data is on file it may be manipulated in a variety of ways. Perhaps the best way to evaluate whether DMS can help you in your business is to look at the data handling options of the latest release. The version under review here is the DMS Diamond, newly available and with the major difference of offering users the ability to utilise data from two files simultaneously.

Creating Your Record Layout

When you first start with DMS you will have to tell DMS and the PET precisely what information you want it to store. You can allocate up to 40 fields (lines) of information, and tell DMS whether those fields should hold numeric, date, or text information. Instructions are simple enough to follow, and a typical file definition to hold a customer mailing list could be something like!

DMS FILE NAME — CLIENTS File Title = Customer records

Created 01/APR/82 Last accessed 15/JUN/82
Records used + 78

Company	1	C	35	Address 1	2	C	30
Address 2	3	C	30	Address 3	4	C	30
Address 4	5	C	30	Contact	6	C	30
Salutation	7	C	30	Co. Type	8	C	30
Last Bought	9	D	6	Last mailed	10	D	6
Purchases	11	C	250	Comments	12	C	100
Rep	13	C	10	Action req	14	C	100

Up to 100 characters may be allocated for each record. This is a longer record length than other programs of this type (Silicon Office, for instance, has a maximum record length of 250 characters per record). This means that quite a lot of detail can be stored for each item, person or company on file. DMS can create as many different data files as required using just the one program disk, which is a lot easire than some other packages of this type.

Data Entry

Records may be entered via a standard screen display, or via specially formatted screens drawn up by the user. This is a particularly useful option in that the order of the fields may be changed, fields missed out completely, longer operator prompts inserted, calculations automatically performed etc.

This option is called 'MASK' on the main menu, presumably because it superimposes a mask over the standard file definition. As well as being useful for data entry, this option also allows for recall, amendment and printing of records. Records may be added to the file at any time.

Searching The File

Although individual records can be recalled almost instantly, the fundamental use of a computerised filing system is the recall of batches of records that meet various selection parameters. These selection parameters (or search criteria), are sets of instructions defined by the user. For instance, if DMS is being used for personnel records, your search criteria could be all staff who are:-

a) over a certain age
b) not in the pension scheme
c) on a salary over 9000 pounds

Or in a customer mailing situation you may want to find:-

a) all the customers who inquired into product Y
b) haven't previously bought
c) bought X and not Y
d) have not been mailed for the last two months
e) live in the London area

As a final example, in an equipment servicing schedule, to find all the equipment where the machines:-

a) are due for servicing on or before a certain date
b) are in location X

Up to eight selection parameters are usable at a time, and any number of different sets may be used. Each selection parameter may be connected with 'and' or 'or' to the others, and DEM will search for information equal to, not equal to, or in a range of certain information. There is no need to allocate separate fields for each bit of information as DMS does 'free text searching', so long descriptive fields can be scanned and records found, even if the crucial information is embedded in a long text field.

PETSPEED

FAST ENOUGH FOR THE HUMAN RACE

Our alien won't hang around for slow software. He wants crisp responses and really fast processing.

For the human race too, slow PET BASIC is not good enough. When we run a program, whatever it is, we want fast efficient action, on the bounce.

Petspeed can make any BASIC program run up to 40 times faster, including disk handling. We guarantee that PETSPEED is easier to use and generates faster code than any other PET BASIC compiler for Commodore Systems.

Using PETSPEED is about as simple as it could be. Just type in the name of your program, wait a few minutes and then watch your software run up to 40 times faster.

Petspeed is not simply a compiler, it contains a powerful OPTIMISER. While PETSPEED is compiling, it breaks your program down into tiny fragments and reassembles it removing the unnecessary and simplifying the complex.

Petspeed is fully compatible with PET BASIC and can compile any program.

Also available INTEGER BASIC COMPILER – 150 to 200 times the speed of Basic. Integer Basic is for those applications where the speed of the machine is required without the inconvenience of assembly level programming. Ideal for scientific and educational users. Compatible with Petspeed.

PETSPEED (8000 or 4000 series) £240
Also available for Commodore 64 & 720 machines

INTEGER BASIC (8000 or 4000 series) £165

SPECIAL OFFER: Petspeed PLUS Integer Basic for £320

Prices do not include VAT or postage and packing.

Write or phone today for a free copy of the 'Life in the fast lane' demonstration disk.

Oxford Computer Systems (Software) Ltd
The Old Signal Box, Hensington Road, Woodstock, Oxford OX7 1JR
Telephone: Woodstock (0993) 812700

Software Review

For numeric information, these search criteria are extended to less than, less or equal to, greater than, etc. DMS also has a true date searching option, to find either all dates between two dates, or all dates before or after certain ates, using the whole or just part of the date.

Sets of search criteria can be stored on disk, and then the relevant records used either for printing lists, labels, merging with standard letters, processing in various ways, or merging with other data items.

Sorting

DMS will sort records into order. It sorts either into alphabetic order, numeric order or date order, or combinations ofthese. Sorts are rapid: between 100 and 200 records on a floppy system, and up to 2 or 3 times faster using a hard disk.

Printing Time

Either the whole file or information may be printed, or you can use pre-selected, and/or pre-sorted batches of records. Any print format can be produced. As the tailored reporting option is fairly difficult to learn compared with the rest of the program, it is nice to find a powerful 'standard' reporting facility, which allows for tabular printouts, field ommission, totals etc.

Most of DMS seems to be easy enough to use without a manual, and there is a tendency to be lazy about manual usage. However, reference is vital for the production of tailored reports.

The letter writing option was much easier to get to grips with. Text can be typed directly onto the screen, and information from the records automatically merged as the records are being printed. For anyone wanting a selective mailing program, DMS seems ideal. Either continuous or single sheet stationary can be used, with or without a cut-sheet feeder. Any number of letters, up to three screen fulls in length, can be created by DMS and stored on disk.

Self adhesive labels can also be printed, using the same names and addresses. Although personalised letters are a strong feature of DMS, many commercial users, such as mailing list companies dealing with vast numbers of enquiries, could simply use it to search the files and print the labels.

As well as all the print options, DMS will show the records on screen, count the records which meet the selection parameters, and print or display the totals.

Links to Other Programs

DMS also has, as standard, links to various PET wordprocessing packages such as Wordcraft, Wordpro and Superscript. These were originally offered to complete the 'select and mail' system, now dealt with by DMS as a stand alone package. Other links include one toVisicalc, useful for statistical analysis of data, and there is an open-ended link to user written software or other commercial packages using sequential files. One final one currently under review is a link to the much-praised Pegasus accounting package.

Multi File Referencing

The greatest difference between DMS Diamond and earlier releases is the ability for Diamond users to access two files simultaneously. This means that standard information stores on one file can be drawn into another file on a record by record basis: a useful facility for anyone using fairly static information in other more lively files, such as invoicing etc.

A similar facility is offered to people who have sets of numeric information to be introduced into records for computations or for textual insert. This is useful for exchange rate analysis, discount structures, prices etc., or wherever numeric rates apply.

Mathematical Processing

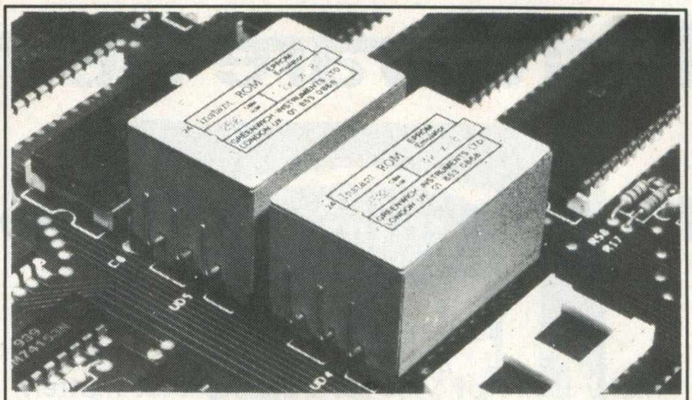
Either the whole file, or pre-selected groups of records, can be processed in a variety of ways, as DMS has a full commercial calculation program as part of the suite. Fields of numeric information can be totalled, sub-totals produced,fields may be added, substracted, multiplied or divided with each other or with constants. These functions can be used for batch price increases, salary changes, discounts, commissions, VAT totalling, sales figures, etc. Thus useful management information and projections can be produced.

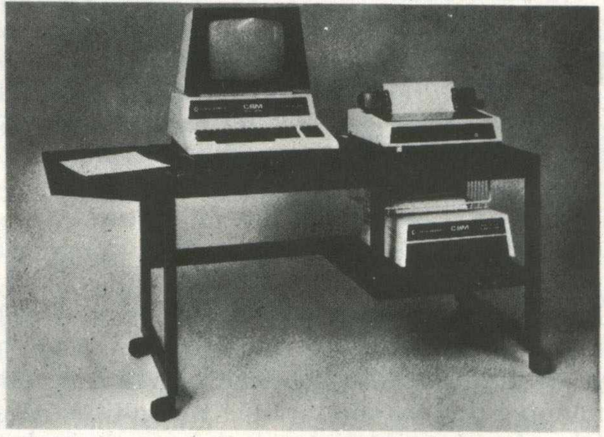
Conclusion

The DMS Diamond, now offering multi-file referencing capabilities, looks to have overcome the major drawback of the 1981 release of DMS. This, plus a new facility for changing the search criteria quickly, makes DMS a powerful and comprehensive package, and useful for a number of different applications.

Compsoft have succeeded in maintaining a simple and straightforward program for the first time computer user, without loss of flexibility.

For further information Compsoft can be reached at Hallams Court, Shamley Green, Near Guildford in Surrey, or by telephone on Guildford (0483) 898545.





Hardware Review

Mini Digital Cassette Recorder

Many users of Vics (or Pets for that matter), who have been using cassette decks as a storage medium, must have yearned for the speed and capacity of disk drives.

Of those, almost as many must have given up in despair at the relatively high price of these units compared to the basic computer. The Vic disk drive, for instance, retails at just under 400 pounds, in other words almost double the price of the Vic itself!

This is clearly a ludicrous situation, but one which regrettably has had no straightforward solution. Tapes are slow, but cheap, and disk drives are fast, but expensive. However, to the rescue of all those who want to store more, and retrieve it faster, Currah Computor Components in Cleveland (tel. 0429 72996 for further information) have come along with their Currah 220M Digital Recorder, or to you and me a digital recorder that uses mini cassettes. There are versions of the 220M for 3000/4000/8000 series Pets, and for the Vic 20. Here we take a look at the Vic 20 model.

Appearance

Considering that the cassettes used are a mere 55 millimetres by 32 millimetres, the unit looks rather large in appearance when placed next to the Vic, measuring in at 230 x 220 x 110 millimetres. Nonetheless it does fit in ergonomically with the rest of the Vic hardware range.

One major complaint is that there is no ON/OFF switch anywhere on the machine: this action has to be performed from the mains, which is rather annoying if you've spent hours trying to work out which plug is which, and then switched the wrong one off! Still, it does arrive with a plug fitted: more than can be said of many other manufacturers.

Also included in the price of 129.99 pounds (excluding our old friend VAT), is post and packing, a free cassette, the aforementioned plug, and a small, but excellent, manual.

First Steps

Most important of all, make sure that the 220M is switched on before attempting to initiate the television/Vic switching on sequence. If you don't, your Vic will power on all right, but the display will look rather alarming, with the familiar blue letters boasting 'Commodore Basic' etc. being replaced by multicoloured ones, and one or two disappearing altogether.

Connection to the Vic is via the memory expansion port, the manual procaliming that you make the connection with the Currah logo uppermost on the connector. Well, unless their logo has been changed to the words 'Vic 20' you won't find one! Having made the connection, a single SYS command produces the words CURRAH CTOS on the screen, and the 220M is ready for action.

Incidentally, if you already have cartridges etc. fitted to the memory expansion, you need to get into expansion unit mode (Commodore, Arfon etc.), whence it will work quite happily.

The first thing the manual (quite rightly) recommends that you do is to check out the unit, making sure that it is installed correctly, and working to your satisfaction. A number of simple Peek and Poke commands suffice for a rough check, but these are mostly designed to ensure that the physical workings of the tape drive are operating properly.

To get a better guide to the actual unit's performance, an introduction to CTOS Basic is then given.

Storage

Having discovered that there will be an error reported if you attempt to rewind a cassette without a cassette present, one gets onto more esoteric subjects such as formatting a directory for a data tape that will use files of length 256 bytes.

'They're Kinda Cute Once You Get Used to Them.'

It is recommended that data tapes and program tapes are kept separately, as the cassette data file pointers would become corrupted, which would in turn corrupt any programs that might also be present on the tape.

Data files can be kept in multiples of 256 bytes, up to a maximum of 1280 bytes (the equivalent in 220M terms of 182 numeric variables), and it is this which determines how much information we can store on our mini-cassette. At the largest size we can have up to 40 files stored on one side of the tape, giving us a storage capacity of 50K. Of course, being nothing more wonderful than an ordinary cassette tape we can use both sides, to end up with a total capacity of 100K per tape. As mentioned, Currah do provide you with a free tape in the price.

In terms of storing programs, it is best to keep to programs that are a minimum of 200 bytes long (not too difficult to achieve), as the CTOS language cannot handle the extremely slight movement of the tape required to save such a small program.

The format of the directory is rather simple: it just tells you the equivalent of the disk header, and the names of any programs that are on there. Another reason for keeping programs and data on separate tapes! To find out how much room you have left on any one tape is not presented to you automatically, but is easy enough to discover. Nor are you told how long any particular file is: rather, you have to find this out for yourself.

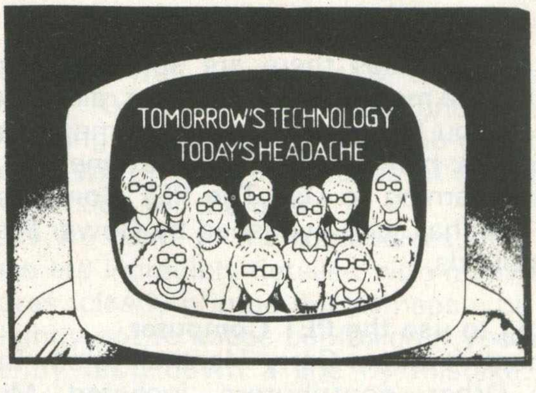
New Language

Currah CTOS adds an additional 21 commands to the existing Vic Basic, all designed to be used in conjunction with the 220M. In other words, you donb't get commands repeated that can be found elsewhere, such has Help, Renumber, and all the other usual additions.

All are prefixed by the '@' symbol, and have been designed to be as easy to remember as possible. Thus a directory load becomes @DL, verify becomes @VE, and so on.

Those of you used to tapes (and to a lesser extent disks) will be pleased by the relative (sorry!) ease with which data can be stored and retrieved. Very simple commands allow storage of variables, both numeric and alpha, and calling these back can (interestingly) be in any order. I.E. if you've saved, say, two alpha and three numeric variables in that order, you do not have to read back the two alpha first if it's the numeric you're after: you can just go straight to the numeric. A type of random sequential filing system I suppose!

A number of useful routines exist within these commands (append a basic program to one



already in memory for instance), and in describing them the manual in turn gives some further extremely useful information for use when using the 220M: how string variables are stored, Pokeing in data, saving machine code, and so on;.

Commands such as read block, write block etc., mean that sequential file handline is now (inexpensively) within our grasp. Sample program listings included cover this, and we can find out about such things as creating files and inputting data, accessing files and retrieving that data, storing names and telephone numbers (now there's an unusual example!), saving and reloading screens of text, all of which are reasonably straightforward.

Various built-in error trapping routines and commands complete a versatile package.

Summary

The Currah 220M Digital Recorder is an efficient, low cost entry into the world of faster storage media. At a price of under 130 pounds it will be welcomed by all those who would previously have baulked at purchasing a disk drive, but would love to make the next step up from cassettes.

If we've not been overly technical here it's because we believe that for this sort of unit you want to know if it's value for money (which it is), and if it competently performs the tasks expected of it (which it does), rather than knowing how many millimetres of tape are used per K of storage. If you want to find that out you can always ring the number mentioned earlier: Currah are a very helpful company, and at this point I'd like to thank them for the loan of the machine, and the help and advice offered along the way.

In the end of course we're down to the old question of you pay your money and you take your choice. If you can afford a disk drive I would not recommend you buying this unit. On the other hand, if you (or your bank manager) can't make that step just yet, but cassette decks are driving you around the bend, this is certainly worth looking at.

To conclude in a single sentence, this is very good value for money.

Book Review

At a rough guess there are some fifty million books on the market at the moment, all purporting to teach you how to use a Pet. Some are better than others: here we take a look at one that isn't, namely Learning to Use The PET Computer, by Garry Marshall, and published by Gower Press at 6.45 pounds.

Learning to use the PET Computer

One mustn't blame Garry Marshall totally for this book. Other contributors included Michael Fluskey, of Gower itself, and Peter Wayth, headmaster of a school in North London.

In his foreword, Mr. Marshall states that really basic, introductory books for the Pet are in short supply, and he is quite right. There is certainly a demand for a good, down to earth, book that takes you all the way from switching the machine on, to programming the beast in machine code.

Books such as Ray West's Programming the Pet/CBM, and the Pet/CBM Personal Computer Guide by Adam Osborne and Carroll Donahue, are probably the best that there are available in this field at present, but experiments with total newcomers to Pets (and perhaps more importantly computers and computing in general) convince me that both those publications tend to leave people far behind.

Don't get me wrong, they are both excellent books, but there is nothing for the complete tyro.

So Mr. Marshall is quite right. However, having stated his aims it is a shame that Learning to use the Pet Computer has ended up the way it has. It is a slim volume, coming in at just 87 pages, which immediately compares unfavourably with the two mentioned earlier, both of which hover around the 500 page mark.

Given 87 pages, one then wonders why so many of them are devoted to an introduction to the history of the Pet, and a very airy-fairy overview of Pets and their uses. Fascinating stuff no doubt, but it certainly didn't tell me how to use a Pet computer.

Indices, forewards, etc. leave us with a bare 58 pages: certainly not enough. The appendices leave out far more than they put in, and are the briefest of brief overviews of the software and hardware scene. Far better to have left this out, than to have bothered wasting space that could be used to much greater advantage.

Content

When you actually get into the meat of the book, it certainly does a lot of jumping about. Commands are introduced at random, with little or no explanation, and many of the program listings perform no useful purpose.

In its defence, the book has got a number of things right. For instance, the program listings are clear in the extreme, and they've adopted the standard put forward by Computing Today (is this why he states in the back of the book that 'I consider this the best of the popular computing magazines ?) for explaining the various graphic symbols that appear when cursoring left, reversing field etc.

Thus, what listings there are are pretty easy to follow. I just wish they did something!

Again in defence the book does give a glance in the direction of most of the things that can be done on a Pet. Graphics, business programming, a quite good section on flow-charting before diving off to produce some code, and indeed a (very!) brief look at some of the special Pet features: user port, memory maps, internal timings etc., but again these are so brief as to be not worth bothering with.

Summary

It is a clearly put together book, well illustrated, well laid out and designed, however ...! To quote Groucho Marx, 'It was one of those books that, once you'd put it down, you just couldn't pick it up again'.

I don't like being this damning of a book, particularly one whose aims are as good as this ones are. Unfortunately, it really doesn't work at all. Too little time is spent explaining just how to 'Learn How to Use a Pet Computer': you certainly wouldn't learn how to program from reading this book.

It is supposedly part of a series: the idea is there, it is a good one, and let us hope that the rest of this series improves on this beginning. One final point: don't charge 6.45 pounds for an 87 page book. Just about every computer magazine on the market is bigger than that, so I don't really think it's justified.

PET/CBM Personal Computer Guide

Since the explosion of the microcomputer, hundreds of books have been brought out for the top computers: Apple, PET/BCM and Tandy. Obviously they differ in quality and quantity, some are brilliant where as some should really never have been written. This one, PET/CBM Personal Computer Guide, is in the former category.

This is the second edition and is an update on the original, covering the newer Commodore Computers; the 4000 and 8000 series PETs. This edition, written by Adam Osborne and Carroll S Donahue (the authors of the original book), gives an indepth coverage of cassette drives, floppy disk drives (both the 2040 and 8050 models), two printers, the 2022 and 2023 models, and a much expnaded tutorial on the Basic language.

The book gives the beginner 500 pages of highly informative information on the computers themselves plus peripherals as well as recent (for 1980) operating software: Basic 4.0 and the Disk Operating System DOS 2.1 and DOS 2.5. Everything the serious CBM computer user wants to know can be found in this book, described in minute detail from a clear and consise introdution on the original PET through to extensive information on editing functions on the 8000 series systems.

Chapters One to Four : The Background chapters

The first three chapters are concerned mainly with the basics of the CBM range (and covering everything in the first book), ie., 2001/8K, 8N, 16N and 2001N/32N, prior to introducing you to the new 4000 and 8000 series cassette units and tapes, disk drives and floppy disks and the printers. The second chapter introduces you to immediate mode: using the PET as a calculator, and then into program mode, etc. Chapter three is all about screen editing: some of the information here had been covered in the first book, but it is essential to repeat it if the user is going to fully understand the update from Basic 2.0 to Basic 4.0 and the screen editing functions involving Basic 4.0 with the 12 inch monitor Pets.

The first three chapters having got you well acquainted with the PET and its basic functions, chapter four begins to teach you how to program on the PET: something it does concisely and very extensively, leaving nothing out. Fifty pages are dedicated to this task, with sub-chapters on Elements of a Programming Language, Basic Statements; Subroutine statements and PEEK and POKE statements being just a few of these, and a Functions sub-chapter covering Arithmetic, String, System and User Defined functions.

Making the Most of CBM Features

The heading explains itself, and here nearly one hundred pages have been given over to indepth instruction on Hardware Features, String Concatenation, Input and Output programming, mathematical programming, graphics and Random Numbers. Again they have left nothing at all out.

A chapter on peripheral devices describing their various uses and how to get the most out of them is given approximately one hundred pages. Yet again this is comprehensively presented. Do this team ever make mistakes?!

The final two chapters cover System information and CBM Basic. Such topics as Memory Maps and Basic Statement Storage are given coverage, as well as Assembly Language Programming. Basic statements, Functions and 8000 Editing Functions are given sub-chapters of their own with nearly fifty pages of information and instruction for the user.

Finally we come to the Appendices.

These are incredibly detailed with many tables of figures, clearly printed but perhaps a bit small and slightly apt to cause confusion if you should accidently drop down a line by mistake: book publishers should provide magnifying glasses!

Summary

Here is a book so well thought out that you should not need any other to teach you about the PET. It costs a mere £10.95, and for a book that covers all it does, in such depth, £10.95 is very good value for money. The contents are well set out and clearly defined; the print is clear and so are the diagrams; the listings are not too long and are a direct copy from the machine, thus giving less chance of error.

The authors have put this book together extremely carefully, and have taken great care not to skip about from one subject to another. As far as I am concerned, unless you're the kind of person who's going to spend the rest of their lives locked up in a garret, programming in machine code and designing ever more complicated boards and circuits, this is an extremely useful guide to becoming a competent PET/CBM user.

It is printed by Osborne/McGraw-Hill in Berkelye, California, but is readily available from all top computer book shops.

Guest Expert

An old ROM version of the Commodore DOS Support Program

In my last article I described a machine code routine to enable loading and saving disk programs when using an 8050 Disk Drive in conjunction with an Old ROM PET. Although the system works well in the described form, it requires the additional OPEN and SYS commands which are not necessary when using the later ROM PETs. When Commodore introduced the Disk Drives they also produced a DOS Support Program which simplified the Load and Save commands, allowed a simple method of passing commands direct to the Disk and displaying the contents of the Disk directories. In its published form, the DOS Support Program would only run on later ROM PETs as it accessed various routines in the Basic interpreter. As I wished to take advantage of the above facilities when using an 8050 Disk Drive on my Old ROM PET, I converted the published program and the following are details of the final result. It must be pointed out that the converted routine will only run on Old ROMPETs and the Disk Load/Save routine described in the previous article, must already be loaded into Cassette Buffer 2.

The DOS Support program

The Commodore DOS Support Program consists of four component parts:
(i) An instructional Basic component to display program operating details.
(ii) The actual DOS Support Program in machine code.
(iii) A relocation routine to place the DOS Support Program at the top of available store and adjust the top of the store pointer.
(iv) A universal Wedge routine to couple into the normal Basic Interpreter routines.

In its currently issued form, the DOS Support Program is issued as a Basic program with item (i) above written in Basic and followed by items (ii), (iii) and (iv) in machine code but contained within the area of the whole Basic program, i.e. the End of Program/Start of variable table pointers are set to after item (iv) above. As will be appreciated, the number of changes necessary to run on the Old ROM PET, were considerable and in order to allow an easy presentation in this article, the program is shown as a complete Basic program. The machine code is contained in Data statements for poking direct into their final locations. Figure 1 shows the complete Basic program and the user is advised, after input, to save the program on Disk (using the Load/Save routine in my previous article) or tape before running, as one mistake in the Data statements will cause the PET to crash.

Details of the Basic DOS Program

Referring to Figure 2, lines 5 to 270 contain the instructional lines displayed once the DOS Support Program has been loaded into the top of available store. Lines 510 to 580 contain the Data statements for relocation and Wedge routines, whilest lines 1010 to 1330 are the actual DOS Support Program Data statements. It will be noted that the relocation and Wedge routines are loaded into locations 634 to 752 which are within the Cassette 1 Buffer. This is no problem as, after setting up the DOS Support Program, this code is no longer needed. Figure 2 is a disassembled print of the relocation and Wedge routines when loaded into Cassette 2 Buffer. Figure 3 is a disassembled print of the actual DOS Support Program shown loaded into the top of an 8K RAM although the program is relocatable to the top of any available RAM i.e. there could be another routine above the DOS Support Program. For those who are interested in more details of the code employed in the above routines, reference should be made to Commodore PET Users Club Newsletter, Volume 2, Issue No. 1 which contains full details of the standard DOS Support Program. The prime changes between the Standard and Old ROM versions are in Basic ROM addresses.

Using the Old ROM Version of the DOS Support Program

All the facilities offered in the standard PET DOS Support Version 4.0 are available in this Old ROM version with the addition of a Save program to disk command. In offering the Save program to disk facility the command character @ is used and therefore it is no longer an alternative to which was introduced for business keyboard PETs.

Example. (to save a program)
type: @PROG where PROG is the name of the program to be saved.

The operating instructions for the Old ROM DOS Support Program are displayed on the screen when it is run and are also contained in the Commodore Disk Manual (with the exception of the Save facility described above). The constraints on using INPUT and GET commands in conjunction with a DOS Support Program (referred to in some Commodore publications) do not apply to the Old ROM version. The standard Commodore DOS Support Program includes the ability to print the Disk Directory direct to a Commodore printer. This code has been converted and included in the Old ROM Version but never tested as I do not have access to a Commodore printer. I would like to hear from any user of this facility in this Old ROM Verison just to set my mind at rest that it works.

Using other than 8050 Commodore Disk Units

Although the Old ROM DOS Support Program and the Load/Save routine in my previous article, have only been tested on an 8K Old ROM PET using the 8050 Disk drive, I see no reason why the routines should not work on other Commodore Disk units provided they use the standard Commodore IEEE interface as do the 2040, 3040, 4040 and the latest single drive unit 2031. As thePET has no knowledge of the disk unit type attached and only produces character strings which are sent over the IEEE interface, I see no reason why there should be any compatability problems. I would like to hear from any reader who has the facilities to try alternative units.

```
5 PRINT"□DDDDDDDDDIIIIIIIIRSETTING UP DOS□":GOTO500
10 PRINT" "TAB(11)"          "
20 PRINTTAB(7)"□.PET OLD ROM DOS SUPPORT□"
30 PRINTTAB(14)"NOW  LOADED
40 PRINTTAB(9)"  COMMANDS FOLLOWING"
50 PRINTTAB(10)"A > IN COLUMN 1 WILL"
60 PRINTTAB(9)"BE PASSED TO THE DISK.0"
90 PRINTTAB(7)"CMD        DESCRIPTION"
140 PRINTTAB(7)"$      DIRECTORY BOTH DRIVES
150 PRINTTAB(7)"$0     DIRECTORY DRIVE 0
160 PRINTTAB(7)"$1     DIRECTORY DRIVE 10"
180 PRINTTAB(7)"  ALL DISK COMMANDS MAY BE
190 PRINTTAB(7)"ENTERED AS IF THEY WERE IN
200 PRINTTAB(7)"A PRINT# STATEMENT.
220 PRINTTAB(11)"□SPECIAL COMMANDS
230 PRINTTAB(7)"□/      LOAD A PROGRAM
240 PRINTTAB(7)"↑      RUN  A PROGRAM
245 PRINTTAB(7)"@      SAVE A PROGRAM
250 PRINT"  SPECIAL COMMANDS START IN COL.1 AND
260 PRINT"ARE FOLLOWED BY A DISK FILENAME.
270 NEW
500 FORI=634TO752:READA:POKEI,A:NEXT
510 DATA 165,134,24,233,16,133,134,165,135,233,2,133,135,169,7,141
520 DATA 248,3,133,228,165,134,133,174,165,135,133,175,169,0,133,227
530 DATA 141,247,3,160,1,177,227,145,174,238,247,3,208,3,238,248
540 DATA 3,173,247,3,201,16,208,7,173,248,3,201,9,240,9,200
550 DATA 208,227,230,228,230,175,208,221,165,134,141,247,3,165,135,141
560 DATA 248,3,24,169,203,109,247,3,141,247,3,144,3,238,248,3
570 DATA 169,76,141,246,3,96,169,76,133,194,164,134,166,135,200,208,1
580 DATA 232,132,195,134,196,96
600 SYS(634):B=PEEK(134):C=PEEK(135):D=(C*256)+B
620 FORI=DTOD+528:READA:POKEI,A:NEXT:SYS(736):GOTO10
1010 DATA 234,230,201,208,2,230,202,134,197,186,189,1,1,201,157,208,9
1020 DATA 189,2,1,201,195,208,2,240,5,166,197,76,200,0,165,202
1030 DATA 208,41,165,201,201,10,208,35,160,0,177,201,133,197,200,201
1040 DATA 62,240,14,201,64,240,29,201,47,240,25,201,94,240,21,208
1050 DATA 10,177,201,240,9,201,36,240,7,208,7,76,200,0,240,114
1060 DATA 240,114,208,114,200,177,201,208,251,136,208,3,76,28,206,132
1070 DATA 238,152,56,101,201,133,201,169,11,133,249,169,0,133,250,169
1080 DATA 8,133,241,169,5,133,239,165,197,201,64,240,29,169,96,133
1090 DATA 240,32,45,245,32,246,3,169,96,133,240,165,197,201,94,240
1100 DATA 6,32,255,243,32,34,244,184,80,20,169,97,133,240,32,45
1110 DATA 245,32,246,3,169,97,133,240,32,160,3,76,200,0,164,197
1120 DATA 192,94,240,6,32,58,3,76,139,195,32,62,3,104,104,76
1130 DATA 181,198,240,4,240,107,208,69,132,201,169,8,133,241,32,182
1140 DATA 240,169,111,133,240,32,44,241,165,197,201,62,240,6,32,135
1150 DATA 241,184,80,36,32,135,241,201,13,240,6,32,234,227,184,80
1160 DATA 243,32,234,227,165,197,201,62,208,6,32,122,241,76,200,0
1170 DATA 169,5,32,205,242,76,139,195,201,48,208,219,96,169,8,133
1180 DATA 241,169,111,133,240,32,186,240,165,240,32,44,241,230,201,160
1190 DATA 0,177,201,240,6,32,103,241,184,80,242,32,126,241,76,200
1200 DATA 0,200,177,201,208,251,136,132,238,169,11,133,249,169,0,133
1210 DATA 250,169,8,133,241,165,239,133,197,173,100,2,141,112,2,169
1220 DATA 96,133,240,169,14,133,239,169,63,32,130,241,32,45,245,169
1230 DATA 0,141,12,2,160,3,132,238,162,14,32,139,247,32,207,255
1240 DATA 133,198,172,12,2,208,41,32,207,255,133,199,172,12,2,208
1250 DATA 31,164,238,136,208,224,32,204,255,174,112,2,224,3,240,5
1260 DATA 166,197,32,220,247,166,198,165,199,32,159,220,169,32,208,4
1270 DATA 208,97,208,194,32,210,255,32,204,255,162,14,32,139,247,32
1280 DATA 207,255,72,32,204,255,104,174,12,2,208,71,201,0,240,39
1290 DATA 174,112,2,224,3,240,5,166,197,32,220,247,32,210,255,32
1300 DATA 204,255,32,42,243,240,44,32,228,255,240,206,201,32,208,202
1310 DATA 32,228,255,240,251,208,195,169,13,174,112,2,224,3,240,5
1320 DATA 166,197,32,220,247,32,210,255,32,204,255,32,126,241,160,2
1330 DATA 208,160,104,32,204,255,169,14,32,205,242,104,104,76,139,195
```

```
 ADDRESS   MACHINE       ASSEMBLER
DEC    HEX  CODE            CODE
==================================
 634   27A  A5 86        LDA  $86
 636   27C  18           CLC
 637   27D  E9 10        SBC  #$10
 639   27F  85 86        STA  $86
 641   281  A5 87        LDA  $87
 643   283  E9 02        SBC  #$02
 645   285  85 87        STA  $87
 647   287  A9 07        LDA  #$07
 649   289  8D F8 03     STA  $03F8
 652   28C  85 E4        STA  $E4
 654   28E  A5 86        LDA  $86
 656   290  85 AE        STA  $AE
 658   292  A5 87        LDA  $87
 660   294  85 AF        STA  $AF
 662   296  A9 00        LDA  #$00
 664   298  85 E3        STA  $E3
 666   29A  8D F7 03     STA  $03F7
 669   29D  A0 01        LDY  #$01
 671   29F  B1 E3        LDA  ($E3),Y
 673   2A1  91 AE        STA  ($AE),Y
 675   2A3  EE F7 03     INC  $03F7
 678   2A6  D0 03        BNE  $2AB
 680   2A8  EE F8 03     INC  $03F8
 683   2AB  AD F7 03     LDA  $03F7
 686   2AE  C9 10        CMP  #$10
 688   2B0  D0 07        BNE  $2B9
 690   2B2  AD F8 03     LDA  $03F8
 693   2B5  C9 09        CMP  #$09
 695   2B7  F0 09        BEQ  $2C2
 697   2B9  C8           INY
 698   2BA  D0 E3        BNE  $29F
 700   2BC  E6 E4        INC  $E4
 702   2BE  E6 AF        INC  $AF
 704   2C0  D0 DD        BNE  $29F
 706   2C2  A5 86        LDA  $86
 708   2C4  8D F7 03     STA  $03F7
 711   2C7  A5 87        LDA  $87
 713   2C9  8D F8 03     STA  $03F8
 716   2CC  18           CLC
 717   2CD  A9 CB        LDA  #$CB
 719   2CF  6D F7 03     ADC  $03F7
 722   2D2  8D F7 03     STA  $03F7
 725   2D5  90 03        BCC  $2DA
 727   2D7  EE F8 03     INC  $03F8
 730   2DA  A9 4C        LDA  #$4C
 732   2DC  8D F6 03     STA  $03F6
 735   2DF  60           RTS
 736   2E0  A9 4C        LDA  #$4C
 738   2E2  85 C2        STA  $C2
 740   2E4  A4 86        LDY  $86
 742   2E6  A6 87        LDX  $87
 744   2E8  C8           INY
 745   2E9  D0 01        BNE  $2EC
 747   2EB  E8           INX
 748   2EC  84 C3        STY  $C3
 750   2EE  86 C4        STX  $C4
 752   2F0  60           RTS
 75
```

FIGURE 2 RELOCATION & WEDGE ROUTINE

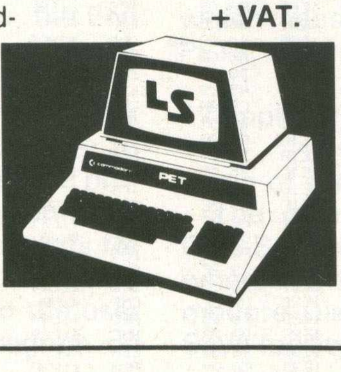
```
 ADDRESS   MACHINE       ASSEMBLER
DEC    HEX  CODE            CODE
=================================
 7663  1DEF  EA         NOP
 7664  1DF0  E6 C9      INC  $C9
 7666  1DF2  D0 02      BNE  $1DF6
 7668  1DF4  E6 CA      INC  $CA
 7670  1DF6  86 C5      STX  $C5
 7672  1DF8  BA         TSX
 7673  1DF9  BD 01 01   LDA  $0101,X
 7676  1DFC  C9 9D      CMP  #$9D
 7678  1DFE  D0 09      BNE  $1E09
 7680  1E00  BD 02 01   LDA  $0102,X
 7683  1E03  C9 C3      CMP  #$C3
 7685  1E05  D0 02      BNE  $1E09
 7687  1E07  F0 05      BEQ  $1E0E
 7689  1E09  A6 C5      LDX  $C5
 7691  1E0B  4C C8 00   JMP  $00C8
 7694  1E0E  A5 CA      LDA  $CA
 7696  1E10  D0 29      BNE  $1E3B
 7698  1E12  A5 C9      LDA  $C9
 7700  1E14  C9 0A      CMP  #$0A
 7702  1E16  D0 23      BNE  $1E3B
 7704  1E18  A0 00      LDY  #$00
 7706  1E1A  B1 C9      LDA  ($C9),Y
 7708  1E1C  85 C5      STA  $C5
 7710  1E1E  C8         INY
 7711  1E1F  C9 3E      CMP  #$3E
 7713  1E21  F0 0E      BEQ  $1E31
 7715  1E23  C9 40      CMP  #$40
 7717  1E25  F0 1D      BEQ  $1E44
 7719  1E27  C9 2F      CMP  #$2F
 7721  1E29  F0 19      BEQ  $1E44
 7723  1E2B  C9 5E      CMP  #$5E
 7725  1E2D  F0 15      BEQ  $1E44
 7727  1E2F  D0 0A      BNE  $1E3B
 7729  1E31  B1 C9      LDA  ($C9),Y
 7731  1E33  F0 09      BEQ  $1E3E
 7733  1E35  C9 24      CMP  #$24
 7735  1E37  F0 07      BEQ  $1E40
 7737  1E39  D0 07      BNE  $1E42
 7739  1E3B  4C C8 00   JMP  $00C8
 7742  1E3E  F0 72      BEQ  $1EB2
 7744  1E40  F0 72      BEQ  $1EB4
 7746  1E42  D0 72      BNE  $1EB6
 7748  1E44  C8         INY
 7749  1E45  B1 C9      LDA  ($C9),Y
 7751  1E47  D0 FB      BNE  $1E44
 7753  1E49  88         DEY
 7754  1E4A  D0 03      BNE  $1E4F
 7756  1E4C  4C 1C CE   JMP  $CE1C
 7759  1E4F  84 EE      STY  $EE
 7761  1E51  98         TYA
 7762  1E52  38         SEC
 7763  1E53  65 C9      ADC  $C9
 7765  1E55  85 C9      STA  $C9
 7767  1E57  A9 0B      LDA  #$0B
 7769  1E59  85 F9      STA  $F9
 7771  1E5B  A9 00      LDA  #$00
 7773  1E5D  85 FA      STA  $FA
 7775  1E5F  A9 08      LDA  #$08
 7777  1E61  85 F1      STA  $F1
 7779  1E63  A9 05      LDA  #$05
 7781  1E65  85 EF      STA  $EF
 7783  1E67  A5 C5      LDA  $C5
 7785  1E69  C9 40      CMP  #$40
 7787  1E6B  F0 1D      BEQ  $1E8A
 7789  1E6D  A9 60      LDA  #$60
 7791  1E6F  85 F0      STA  $F0
 7793  1E71  20 2D F5   JSR  $F52D
 7796  1E74  20 F6 03   JSR  $03F6
 7799  1E77  A9 60      LDA  #$60
 7801  1E79  85 F0      STA  $F0
 7803  1E7B  A5 C5      LDA  $C5
 7805  1E7D  C9 5E      CMP  #$5E
 7807  1E7F  F0 06      BEQ  $1E87
 7809  1E81  20 FF F3   JSR  $F3FF
 7812  1E84  20 22 F4   JSR  $F422
 7815  1E87  B8         CLV
 7816  1E88  50 14      BVC  $1E9E
 7818  1E8A  A9 61      LDA  #$61
 7820  1E8C  85 F0      STA  $F0
 7822  1E8E  20 2D F5   JSR  $F52D
 7825  1E91  20 F6 03   JSR  $03F6
 7828  1E94  A9 61      LDA  #$61
 7830  1E96  85 F0      STA  $F0
```

Guest Expert

```
7832  1E98  20 A0 03  JSR  $03A0
7835  1E9B  4C C8 00  JMP  $00C8
7838  1E9E  A4 C5     LDY  $C5
7840  1EA0  C0 5E     CPY  #$5E
7842  1EA2  F0 06     BEQ  $1EAA
7844  1EA4  20 3A 03  JSR  $033A
7847  1EA7  4C 8B C3  JMP  $C38B
7850  1EAA  20 3E 03  JSR  $033E
7853  1EAD  68        PLA
7854  1EAE  68        PLA
7855  1EAF  4C B5 C6  JMP  $C6B5
7858  1EB2  F0 04     BEQ  $1EB8
7860  1EB4  F0 6B     BEQ  $1F21
7862  1EB6  D0 45     BNE  $1EFD
7864  1EB8  84 C9     STY  $C9
7866  1EBA  A9 08     LDA  #$08
7868  1EBC  85 F1     STA  $F1
7870  1EBE  20 B6 F0  JSR  $F0B6
7873  1EC1  A9 6F     LDA  #$6F
7875  1EC3  85 F0     STA  $F0
7877  1EC5  20 2C F1  JSR  $F12C
7880  1EC8  A5 C5     LDA  $C5
7882  1ECA  C9 3E     CMP  #$3E
7884  1ECC  F0 06     BEQ  $1ED4
7886  1ECE  20 87 F1  JSR  $F187
7889  1ED1  B8        CLV
7892  1ED4  20 87 F1  JSR  $F187
7895  1ED7  C9 0D     CMP  #$0D
7897  1ED9  F0 06     BEQ  $1EE1
7899  1EDB  20 EA E3  JSR  $E3EA
7902  1EDE  B8        CLV
7903  1EDF  50 F3     BVC  $1ED4
7905  1EE1  20 EA E3  JSR  $E3EA
7908  1EE4  A5 C5     LDA  $C5
7910  1EE6  C9 3E     CMP  #$3E
7912  1EE8  D0 06     BNE  $1EF0
7914  1EEA  20 7A F1  JSR  $F17A
7917  1EED  4C C8 00  JMP  $00C8
7920  1EF0  A9 05     LDA  #$05
7922  1EF2  20 CD F2  JSR  $F2CD
7925  1EF5  4C 8B C3  JMP  $C38B
7928  1EF8  C9 30     CMP  #$30
7930  1EFA  D0 DB     BNE  $1ED7
7932  1EFC  60        RTS
7933  1EFD  A9 08     LDA  #$08
7935  1EFF  85 F1     STA  $F1
7937  1F01  A9 6F     LDA  #$6F
7939  1F03  85 F0     STA  $F0
7941  1F05  20 BA F0  JSR  $F0BA
7944  1F08  A5 F0     LDA  $F0
7946  1F0A  20 2C F1  JSR  $F12C
7949  1F0D  E6 C9     INC  $C9
7951  1F0F  A0 00     LDY  #$00
```

```
7953  1F11  B1 C9     LDA  ($C9),Y
7955  1F13  F0 06     BEQ  $1F1B
7957  1F15  20 67 F1  JSR  $F167
7960  1F18  B8        CLV
7961  1F19  50 F2     BVC  $1F0D
7963  1F1B  20 7E F1  JSR  $F17E
7966  1F1E  4C C8 00  JMP  $00C8
7969  1F21  C8        INY
7970  1F22  B1 C9     LDA  ($C9),Y
7972  1F24  D0 FB     BNE  $1F21
7974  1F26  88        DEY
7975  1F27  84 EE     STY  $EE
7977  1F29  A9 0B     LDA  #$0B
7979  1F2B  85 F9     STA  $F9
7981  1F2D  A9 00     LDA  #$00
7983  1F2F  85 FA     STA  $FA
7985  1F31  A9 08     LDA  #$08
7987  1F33  85 F1     STA  $F1
7989  1F35  A5 EF     LDA  $EF
7991  1F37  85 C5     STA  $C5
7993  1F39  AD 64 02  LDA  $0264
7996  1F3C  8D 70 02  STA  $0270
7999  1F3F  A9 60     LDA  #$60
8001  1F41  85 F0     STA  $F0
8003  1F43  A9 0E     LDA  #$0E
8005  1F45  85 EF     STA  $EF
8007  1F47  A9 3F     LDA  #$3F
8009  1F49  20 82 F1  JSR  $F182
8012  1F4C  20 2D F5  JSR  $F52D
8015  1F4F  A9 00     LDA  #$00
8017  1F51  8D 0C 02  STA  $020C
8020  1F54  A0 03     LDY  #$03
8022  1F56  84 EE     STY  $EE
8024  1F58  A2 0E     LDX  #$0E
8026  1F5A  20 8B F7  JSR  $F78B
8029  1F5D  20 CF FF  JSR  $FFCF
8032  1F60  85 C6     STA  $C6
8034  1F62  AC 0C 02  LDY  $020C
8037  1F65  D0 29     BNE  $1F90
8039  1F67  20 CF FF  JSR  $FFCF
8042  1F6A  85 C7     STA  $C7
8044  1F6C  AC 0C 02  LDY  $020C
8047  1F6F  D0 1F     BNE  $1F90
8049  1F71  A4 EE     LDY  $EE
8051  1F73  88        DEY
8052  1F74  D0 E0     BNE  $1F56
8054  1F76  20 CC FF  JSR  $FFCC
8057  1F79  AE 70 02  LDX  $0270
8060  1F7C  E0 03     CPX  #$03
8062  1F7E  F0 05     BEQ  $1F85
8064  1F80  A6 C5     LDX  $C5
8066  1F82  20 DC F7  JSR  $F7DC
8069  1F85  A6 C6     LDX  $C6
```

```
8071  1F87  A5 C7     LDA  $C7
8073  1F89  20 9F DC  JSR  $DC9F
8076  1F8C  A9 20     LDA  #$20
8078  1F8E  D0 04     BNE  $1F94
8080  1F90  D0 61     BNE  $1FF3
8082  1F92  D0 C2     BNE  $1F56
8084  1F94  20 D2 FF  JSR  $FFD2
8087  1F97  20 CC FF  JSR  $FFCC
8090  1F9A  A2 0E     LDX  #$0E
8092  1F9C  20 8B F7  JSR  $F78B
8095  1F9F  20 CF FF  JSR  $FFCF
8098  1FA2  48        PHA
8099  1FA3  20 CC FF  JSR  $FFCC
8102  1FA6  68        PLA
8103  1FA7  AE 0C 02  LDX  $020C
8106  1FAA  D0 47     BNE  $1FF3
8108  1FAC  C9 00     CMP  #$00
8110  1FAE  F0 27     BEQ  $1FD7
8112  1FB0  AE 70 02  LDX  $0270
8115  1FB3  E0 03     CPX  #$03
8117  1FB5  F0 05     BEQ  $1FBC
8119  1FB7  A6 C5     LDX  $C5
8121  1FB9  20 DC F7  JSR  $F7DC
8124  1FBC  20 D2 FF  JSR  $FFD2
8127  1FBF  20 CC FF  JSR  $FFCC
8130  1FC2  20 2A F3  JSR  $F32A
8133  1FC5  F0 2C     BEQ  $1FF3
8135  1FC7  20 E4 FF  JSR  $FFE4
8138  1FCA  F0 CE     BEQ  $1F9A
8140  1FCC  C9 20     CMP  #$20
8142  1FCE  D0 CA     BNE  $1F9A
8144  1FD0  20 E4 FF  JSR  $FFE4
8147  1FD3  F0 FB     BEQ  $1FD0
8149  1FD5  D0 C3     BNE  $1F9A
8151  1FD7  A9 0D     LDA  #$0D
8153  1FD9  AE 70 02  LDX  $0270
8156  1FDC  E0 03     CPX  #$03
8158  1FDE  F0 05     BEQ  $1FE5
8160  1FE0  A6 C5     LDX  $C5
8162  1FE2  20 DC F7  JSR  $F7DC
8165  1FE5  20 D2 FF  JSR  $FFD2
8168  1FE8  20 CC FF  JSR  $FFCC
8171  1FEB  20 7E F1  JSR  $F17E
8174  1FEE  A0 02     LDY  #$02
8176  1FF0  D0 A0     BNE  $1F92
8178  1FF2  68        PLA
8179  1FF3  20 CC FF  JSR  $FFCC
8182  1FF6  A9 0E     LDA  #$0E
8184  1FF8  20 CD F2  JSR  $F2CD
8187  1FFB  68        PLA
8188  1FFC  68        PLA
8189  1FFD  4C 8B C3  JMP  $C38B
```

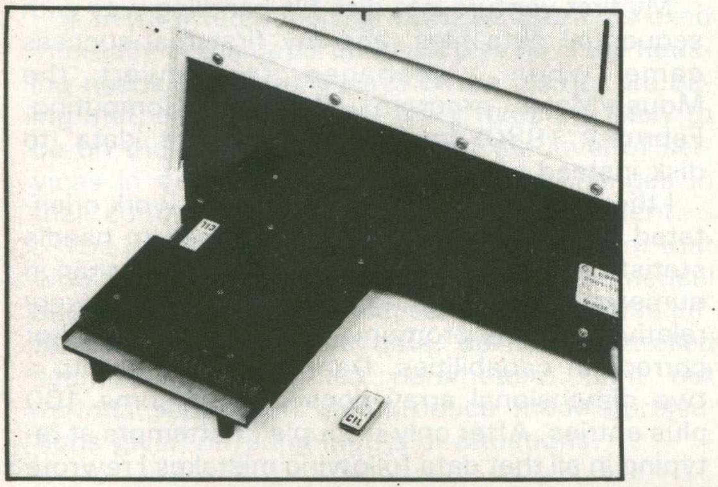


A NOTICE TO ALL

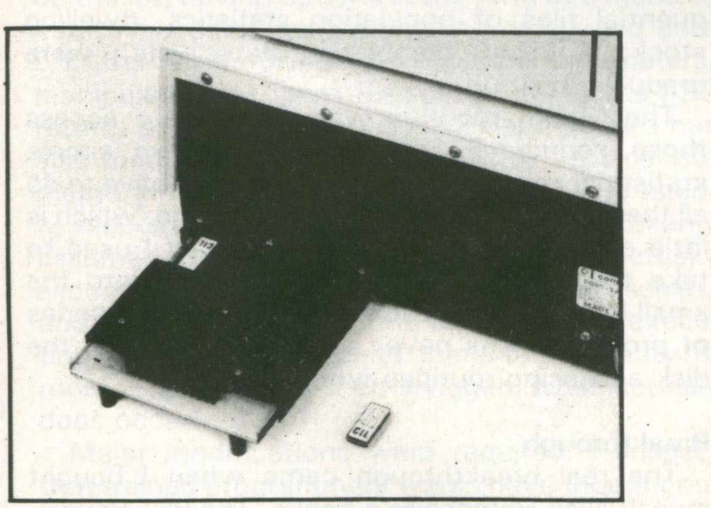
PET LOVERS

INTRODUCE YOUR PETS TO OURS

AND SOLVE YOUR INTERFACE PROBLEMS



Only £195.00



Only £195.00

ANALOGUE/DIGITAL I/O

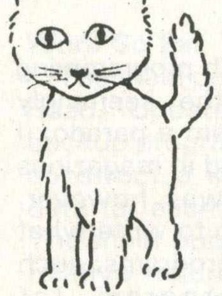
THE PUPI

- **4 ANALOGUE INPUTS (12BIT)**
- **2 ANALOGUE OUTPUTS (12 BIT)**
- **4 RELAY OUTPUTS**
- **4 LOGIC INPUTS**

When connected to the "PET" User Port the PUPI gives you all the above features together with an operating system in EPROM, which interacts with Basic's variables, giving extremely simple operation. Inputs and outputs are ±10V and relays are rated at 10VA. Logic inputs can be used for microswitch sensing etc.

HIGH SPEED A/D CONVERTER

THE PUSSI



- **4 ANALOGUE INPUTS (12 BIT)**
- **50 MICROSECOND CONVERSION**
- **STOP AND START TRIGGERS**
- **DATA ACQUISITION SOFTWARE**

Using an operating system in EPROM, the PUSSI provides a high speed A-D Converter with 4 multiplexed inputs, which is under control of either software, or remote start/stop triggers. A-D Conversion can be carried out from Basic, or Machine Code, with up to 1500 readings entered directly into memory at a software determined rate.

CIL

MICROSYSTEMS LTD

CIL MICROSYSTEMS LTD.
DECOY RD.,
WORTHING,
SUSSEX BN14 8ND.
TELEX: 87515 WISCO G ATTMIC
TEL: (0903) 210474

Write, phone or to obtain further information circle number

A Year in the life of a PET

A piercing scream echoed along the darkened corridor, dying away as quickly and mysteriously as it arose. ''Oh dear'', sighed the typist in a resigned manner. ''That must be the tenth time today his new program has crashed ....''.

Actually it's not been quite as bad as that but I have been pretty near to screaming on many occasions, while on just the odd occasion I even started wondering what life must be like in a monastery far away from printers that won't print, disk drives that won't drive and computers that won't!

I work in a small local authority planning department in what is known at the Local Plans section. Basically my job, in a team of three others with a section head, is to prepare the forward planning policy statements for the district, and to keep track of a fair amount of statistical information — much of this is used in the preparation of ''Local Plans''.

About a year ago we managed to convince our boss and the Council's treasurer of the merit of some form of computer system to handle this data. We ended up with a 3032 computer, 3022 printer and 3040 disk drive.

There was nobody in our department with any programming expertise — we were well and truly on our own.

Programming the Computer

I elected to familiarise myself with programming the computer, and set about the seemingly awesome task. Though it may seem a paradox I found that by typing in games listed in magazines I managed to learn a great deal. It was, however, a process of trial and error. I began to write what now seem incredibly simple little programs: such things as a conversion program for metric/imperial or simple modelling projects where one could input and alter certain parameters to ascertain the net effect in, say, a population projection. Other programs were perhaps outwardly less useful but were of great interest to me, enabling me to become more proficient with screen graphics, presentation etc., or helped me to learn more about the PET and its built-in functions, and increased my knowledge of Basic programming such as string handling, loops etc.

The machine had now been in the office a couple of months and it was becoming increasingly obvious that I had to start producing the 'goods', if only to justify the expenditure! I realised that without a proper knowledge of disk file handling routines I was completely stuck. I was to remain 'stuck' for quite some time — certainly in respect of random access files.

First Venture

My first venture into disk file handling was with sequential data files, and my first real success came when I managed to convert the Mouse/Maze program (Practical Computing, February 1980) for writing the maze 'data' to disk instead of to tape.

I then started to write my first truly work orientated programs. These were designed to handle statistical data to be stored on disk rather than in numerous files littering the office. They were relatively simple programs with very limited error correction capabilities. Data was entered into a two dimensional array consisting of some 400 plus entries. After only a couple of attempts at retyping in all that data following mistakes I rewrote the program to put the data into separate one dimensional arrays, identified by the year, which were then individually written to disk. These sequential files of population statistics, dwelling stock, housing completion rates etc. were gradually built up.

The 'Read' program was intended to access those sequential files and to perform simple statistical calculations. Unfortunately I have to do all the work on the PET in my spare time, which is little enough during the day — in fact I used to take the machine home only to work into the small hours — with the result that this first series of programs was never completed. At least the disk accessing routines worked!

Breakthrough

The real breakthrough came when I bought myself Nick Hampshire's books 'The PET Revealed' and 'The PET Library of Subroutines'. These helped in three main ways:-

a) They gave me an insight into some of the machine's workings and organisation, which though not vital information certainly helped me to understand better what I was doing.

b) They provided very useful routines which I could use in my own programs directly (e.g. the Random Access routines), or indirectly using the machine code routines located at the top of memory to enhance displays etc.

c) They opened up the world of random access to me!

I then started work on a series of programs to store and manipulate a large volume of data relating to the availability of land for residential development. This information, which is kept on ordnance Survey sheets and various schedules, is required for a number of reasons. Perhaps the most demanding and time consuming of these is to complete twice yearly a government return (inown as PS3), which required a search through all of the recorded information to produce sum totals under various categories and headings.

A more interesting use ofthese figures is used internally in the department to plot how the housing needs in different parts of the district are being met, and to see what the effects are likely to be on the population totals and the level of services in various settlements due to changes in their commitments to residential development.

The need was for a data handling system that would manage a fairly large amount of numerical and verbal data, be able to select and analyse different sections of that data, do user controlled searches on specified parameters, print out various schedules, and produce models/projections given differing inputs or situations.

The Start!

I very naively started to write one program to do the lot, having options in the form of a menu to input new data, update existing data, read data off the disk given the search parameters, manipulate data, print schedules, compile a PS3 return, and so on. To make matters worse I has also tried to input all of the selected records into memory from disk for a specific task. I very rapidly ran out of memory space! I had mistakenly assumed that the DIM statement actually allocated the space for the arrays themselves, and when checking for FRE memory after execution of the DIM assumed there to be plenty of room. I have since then twigged what actually does occur!

Major modifications were required. I suspect that trained programmers would have thought out all the requirements of the program quite thoroughly and would have drawn up complex flow charts. Whilst I see the merits in that, and make serious attempts to do just that, the work at the keyboard and the ideas which came while working and running the program soon overtook any flow-charting I had managed. Needless to say I spent many hours, often into the early hours, retyping vase sections of program, often for the most ludicrous of reasons.

The eventual outcome was a suite of programs, sad to say still incomplete for lack of time, which are specific to the major functions listed above. These programs are all linked and will call other programs in the suite on a single key stroke. I have toyed with the idea of using a single set of subroutines and loading only the variable parts of the program, calculating the link address and so on. However, I have chosen to write each program complete with the common subroutines so that I need only alter memory locations 42/43 (decimal). In some future work I think I would like to try the first option to enable variables and arrays to be passed on, but that must wait!

All the programs in the suite were designed with the lay user in mind. They use similar formats and screen presentations. From the programmers' point of view I have tried to be consistent too. They are all structured around the same subroutines, and by and large use the same variable/array names. Line numbers are grouped so that, for instance, all disk read/write routines occur in lines 50000 to 59999 while common routines handling certain display, error trapping and other functions are all from 60000 onwards.

I have even gone to the extent in one case of using the lines 40000 to 49999 for all hard copy output routines because the printer happens to be device number 4. The programs also use a number of machine code routines to draw a border around the 'menu' to flash the prompt message, to print out the screen display to the printer etc. I am now just at the stage of examining ''PET Graphics'' to see whether I can use any of those routines.!

Work So Far

The main programs in the suite are Menu, Input, Read, Update, PS3 and a couple of system backup programs. The input and update programs construct a key file which contains a string of data for each record on the random access disk. The main purpose of this is to keep track of the relative record number. The rest of each string contains extracts of the main data to enable searches to be carried out on a form of 'sieve'' approach.

This allows the user to select a number of parameters which will be looked for in the programs' search routines. The result is a list of records whose data accords with the search parameters. The read program then uses this list to find the relative record number, calculates the track and sector number of that record, and reads it into memory. Depending on the function being used a further check can be made on parts of the data not in the key file to ensure that this record should in fact be used by the program for the designated purpose e.g. printing schedules, giving totals etc.

As you can see the key file is a very important part of the operation of the suite, and so one of the backup programs is designed to read all of the random records in order (from track 1 sector 0) and reconstruct the key file array relevant to each record. Another such program constructs a simple array of all record numbers where the data has become obsolete. This enables re-use of blocks on the disk under program control. I have not used the block-allocate and block-free commands as I write all the random data to the disk using a relative record number — in other words DOS/BAM haven't the foggiest what is on the disk but the program does ..... I hope!

Refinements

programs are gradually being refined. For instance, the input program now searches for the existence of any vacant blocks via the 'Vacbloc' array set up by the backup program. This array holds the record number of obsolete data blocks. Once this array of record numbers has been 'used up' the numbering of further new records reverts to sequential numbering starting from the total number of records on disk. The program simply reads this variable from a counter incremented while the key file arrays are being read in using a simple loop which checks for the end of file status flag.

I hope shortly to introduce a system whereby up to a screenful of user instructions may be called from the disk directly onto the screen and then return to the point in the program the user had reached, without disturbing the memory, variables etc. I feel it ought to be possible and am busy scanning the journals and mags!

Other developments rather than refinements will enable a wide range of statistical analyses to be carried out on the information stored by these programs. This would include changes in the dwelling stock, growth rates of settlements, proportions of local government to private housing ..the list is almost endless (so my section head would have me believe!).

Summary

Perhaps I have been able to give some idea of a layman's progress with the PET, and made even a little encouragement for those who, like me, are beginners and often get that "bogged down" feeling — just persevere!

Upgrade Roms for Old 8k Pets

For those of you with the old 8k PET and 24 pin ROMs who envy the three empty sockets in the newer machines, good news.

The 'upgrade' ROMs for these machines only occupy four of the seven sockets and a simple cut and hack operation on your main board will enable you to use two of the freed sockets. All three sockets may be used by the simple addition of one more IC.

Furthermore, if you want to only use one socket for the toolkit, or the Word Pro 3, you don't even have to pull the board from the case.

A word of advice, however. If you are not reasonably expert in handling this type of operation (soldering directly to the IC pins), or live and work in a high 'static electricity' environment, don't try it.

This modification required two sequences of events:

1. Change the bank select lines to the emptied ROM sockets, and
2. Change the bank access to the external PET data bus.

Both these operations may be done with the main board still in the case if only one socket is to be enabled. If you want two sockets operational, you have to pull the board to get at a trace on the underside.

Change Bank Select Lines

The 'bank' addresses of the three freed sockets has to be changed from C, D, and F, (in hexadecimal notation; 12, 13, and 15, in decimal), to 9, A, and B, or whatever. The three bank select lines of interest originate at IC G2, pin 14 (select C or, 12), pin 15 (select D, or 13), and pin 16 (select F, or 15). They run a short distance toward the front of the board on the underside of the card, then surface near socket H4. They run across upper surface of the board toward the power supply for several inches then return to the underside of the board to connect to pin 20 of the appropriate socket. These three traces are to be cut just above H5. Be very sure that the traces are completely cut and that you remove all the metal scrap that is generated.

Now carefully solder three wires to IC G2 pin 10 (select 9), pin 11 (select A, or 10), and pin 13 (select B, or 11). Run these wires to the solder dots on the ROM ends of the traces just cut. Simple. But if you try to get the machine to recognize ROMs plugged into these sockets, it will insist that there is nothing there!

Data Bus Access

The problem lies in the design of the data bus. The PET presumes that all addresses between the screen memory and the four ROMs of the operating system are external to the machine. When accessing these addresses, it enables the external data bus drivers. These drivers take data from the outside world and place it on the internal bus. In the meantime the ROM you have just installed is trying to do the same thing. That doesn't work well at all. The solution here is quite simple; don't allow the external data bus drivers to be activated when your ROM Socket is being addressed.

The IC which controls this action is G4, a 74LS21. Two of the input lines to this chip are not used in the original model and may be 'stolen' to enable two of the freed sockets. The trace that ties the two pins of interest (pins 4 and 5) together is on the underside of the board. If only one socket is to be used (say for WordPro) you do not have to separate them and the board can be left in place during the alteration.

These pins are held at logic '1' ('high') by a resistor at IC G3. The trace of interest is on the upper surface of the board, and goes from the resistor to IC G3, pin 9, and IC G4, pin 5. Cut the trace near G4, remove the scrap metal, and run a wire from G4 pin 5 to the appropriate bank select wire installed in part 1, above. To use a second socket, you have to remove the main board, cut the trace connecting IC G4 pins 4 and 5 together, and run a second wire from pin 4 to another bank select line.

The third socket may be used, but you have to instal another IC. Drop me a line and I'll send you a schematic. My address is Box 481, Rossland B.C., VOG 1YO.

If you are like me ad have the Toolkit hung on the side of the PET at the expansion port, you can even have two ROMs with the same address, selectable with an external switch. The bank select signal goes to the switch and is routed to the appropriate ROM. The unselected ROM must have the bank select line pulled high with a 1k resistor to the +5 volt power supply line. The circuit is left as an exercise, but don't forget to switch the external data bus drivers at the same time.

Real Sound in Real Time

If you ever wanted to synthesize instruments in real time on your PET/CBM, then this article will tell you of one marvelous, unadvertised, unpromoted program. It is Instrument Synthesis Software Package (K-1001-6C) from Micro Technology Unlimited. The cost of the program is $50. The required digital-to-analog converter (DAC) is $60.

Hal Chamberlin who is considered the father of real-time microcomputer music, has written several eloquent papers on the principles behind the system (see references). He invented an ingenious software method of producing multi-voice sound on the PET and has recently elaborated that invention to include instrument building.

Hence, with minimal cost, a PET user can have an in-house synthesizer comparable in quality to some expensive hardware systhesisers. I recently attended a concert in Philadelphia at which Frank Covitz and Cliff Ashcraft, known in the trade as the Diatonic Duo, demonstrated the MTU's system with help of PET's distinguished 6502 ancestors: the Baroque Aim and the Home-brew Kim. The excellent sounds included a super 17th century harpsichord and a $2 ukelele pitch pipe.

Tiny Bit of History

Originally, the Chamberlin-style four-voice music software produced organ-like sound of varied characteristics, achieved through user's complete control of overtones. The sound had a sudden onset, level sustain and a sudden drop. There exist two such systems: AB Computers' Visible Music Monitor, written by Frank Levinson and known for its superb graphic editor, and MTU's four-voice package written by Frank Covitz, known for its lack of an editor. Both systems are currently available.

The State of the Art

Hal Chamberlin elaborated on the idea of his original system by adding an instrument synthesis feature. It is now possible to synthesize just about any sound, real or imagined. Fairly successful imitation of existing instruments has been done. Invention of new instruments via sound analysis permits us to hear, for example, Bach's inventions played by a word ''NO'' or ''Raindrops Keep Falling on my Head'', played, quite appropriately, by a straw and, I think, water dripping into a glass.

The program permits the user to select from the book and/or create ''instruments'' by specifying the amplitude and its rate of change over time for each harmonic. This sounds hard, but isn't. Entering X-Y (time-amplitude) coordinates does the trick, the program does the rest. The so defined composite amplitude envelope accomplishes the distinctions between various instruments: plucked, struck and blown instruments have been successfully implemented. The necessity for very high overtones does not yet permit building bowed instruments. There is no limit to the variety of sounds you can make (within the 8100 cps sampling rate), including different touch or attack characteristics, as well as crazy combinations of sound effects.

In fact, one of the features of this software I treasure most is that to the extent of available memory and speed of processing, it is infinitely flexible. Many sounds can be made, and any note-timing protocol can be set, not limiting you to a strict observance of the score. Knowledge of sound theory is not needed. The book gives enough information.

MTU has a demonstration audio tape available for $5. You may hear the sounds before deciding to plunge into the synthesis. For those who alreayd have the program, MTU sells a floppy or two with precoded song data. The program and song data are loaded into the PET and, simply, RUN.

The explanation of the physics behind the system and the instructions on how to use the software are provided. I found only one, insignificant, error in the book. The instructions are complete, but difficult to use at first. The reason is that it is not immediately apparent from reading the descriptions of the available commands what is the commands' purpose. Their utility becomes obvious when one studies the book and a coded song together. The syntax of commands is unambiguous and yields an error message or silly sounding music if not observed. Locating an error is easy if the program's extensions are used.

How Does it Sound?

I like the sound. People who are familiar with the earlier, organ-like, music may be interested to know that through several ingenious programming tricks, Frank Covitz has improved the sound quality of the system, even though this program has to perform many more calculations in real time than it did previously. There are no "clicks" between the notes and the signal-to-noise ratio has been increased by an audible 6db.

Chamberlin and Covitz remind the readers that the sounds are "guitar-like", "horn-like" etc. and not quite "the real thing." What's missing is the "liveliness" and the resonances of the instrument when limitations are built (though piano in the lower range is superb, since it uses 26 harmonics!). The reason is that the 8kc rate is too slow to permit high harmonics without running into distortion problems at high pitch. Hence, it is not appropriate to compare it with the real thing. But these tones are nevertheless pleasing and the key characteristics of the imitated instruments are clearly present. Invented instruments, of course, are not subject to such comparison, and, therefore sound just terrific. One of my favourites is a wind instrument that gently whistles while it plays, the whistle coming in a bit after the highest amplitude of the rest of the sound.

Musically, many of the instruments in the book and in the already transcribed songs, are beautiful. Their interesting characteristics and their variety compensate for the "smallness" of PET's sound. Many instruments chosen for a particular piece of music sound appropriate to the coded music, even if that piece of music was written for another instrument. It takes some doing to orchestrate things. I like the results produced by Covitz, Ashcraft and Chamberlin.

Educational Possibilities?

Listening to transcribed music, in many instances, makes the music more accessible to the listener. The impact is similar to, for instance, Segovia transcriptions of Bach's keyboard or violin works for the guitar or the Canadian Brass Quintet's transcriptions of all sorts of music for trumpets and things.

The system has a great entertainment value built in. It may also have an educational value. You can study various aspects of music and the physics of sound without fancy sound analyzing gizmos, and without a multitude of instruments at hand. Your ears and the graphing paper tell you just about the whole story.

It is interesting to hear music performed at different speeds, without changing pitch. It's also interesting to study what happens if a line of music is played backwards or by a variety of different instruments. It points out the tie between an instrument, melody, mood, etc. It can be used to study how instruments fit the music. It shows that to be able to utilize the multitude of possible sound configurations, new music may need to be written. These are some of the things that cannot possible be observed with just a record player or a tape recorder.

A curious result of coding your own music is a realization of a fact, well known to musicians, that the musical score is only a hint and that it is totally inadequate as a means of communicating the composer's wishes (whatever they may have been). Pages of notes are a necessary requirement for music, but by no means sufficient. It takes some doing to tell the PET to not perform like a machine but it can be done since the program places almost no limitations on the user. Your coded music will not sound big and powerful, but it can have a character of its own. Therein lies the fun of putting music on the PET.

How Easy to Code?

It could be worse if the book was ambiguous, which it isn't. But once you read the book, accept the syntax rules and code one or two measures of a song. It's not hard at all. Just tedious.

MTU considers this product an unfinished work. It is not promoted and as far as I can tell it will not be promoted until the "human interface", or a note and instrument entering editor is written. Writing such an editor is a great challenge, where half-way measures won't do. I can't tell when the editor will be available for the PET. Hence, we are stuck with coding instruments and music via the Machine Language Monitor. No machine code knowledge is required to use the system as it is an interpreter specifically designed for doing music. For instance, two of some 16 commands look like this: 'F2 tt' means tempo, 'FE pp bb' means play a segment that is in memory at location pp bb. Knowing how to use PET's Monitor and a sense of pages in its memory is needed, but one can learn by doing.

You have two options. One is to use the program and the song data for enjoying it and wait with coding your own things until the editor is written. The second option is to jump in now and not miss all the fun. The system, even though tedious in places, is useable, error-free and invites experimentation. Debugging extensions to the program have been provided which help locate a note, or a song segment for easy finding of coding errors.

In case you might be worried that your song data may become obsolete should an editor be introduced, I'd say "don't worry". One of the key policies of MTU has been compatibility. It is unlikely they will obsolete anything. Just as, at the present time, a song coded on an APPLE, for instance, will play, **with no modifications** on any one of PET's many releases and 6502 relatives, I am pretty sure the introduction of an editor will not change a thing.

Sound 'n' Vision

System Considerations

MTU strongly recommends a 32K PET. Only several available songs will play in a 16K PET. Neither the program (2.5K) nor the song data (varies. $\frac{1}{4}$K-2K is a good guess) use much memory. The waveforms for the instruments, however, gobble it up pretty fast. A disk drive is not essential, though always very helpful. Monitor extensions, such as the Supermon or Extramon are essential. They permit easier editing of data, specifically, inserting, deleting and transferring code. You can't do without them. Both are public domain programs, available from various sources. The MTUs' Visible Memory board can, optionally, be used to see the elements of an instrument. A digital-to-analog converter board needs to be plugged into the User port of the PET. Two DACs can be used for stereo effect (2 voices to each channel). In a concert hall the stereo effect was incredible, but in a home situation it's not needed (I'm biased: I think monaural records are OK).

Other Information

Making this kind of music on your microcomputer is a lot of fun now and the results can be quite musical, even though the sound quality is that of a slightly noisy AM radio if only quiet instruments are picked. Do not let this discourage you. We can reasonably expect true high fidelity sound in a short time. All it will take is a faster processor and larger memories. It's worth taking the plunge now and be ready for progress.

REFERENCES

(1) Hal Chamberlin, A Sampling of Techniques for Computer Performance in Music, BYTE magazine, September 1977.

(2) Hal Chamberlin, Advanced Real Time Music Synthesis Techniques, BYTE magazine, April 1980.

(3) Hal Chamberlin, Musical Applications of Microprocessors (a big book, 653 pages), $25.

Getting Usable Video Signals from 12" Monitor PET/CBMs

This program works with any "fat" Commodore computer (8032, 12" 4032, 8096, and SuperPET). It re-configures the video controller to produce signals which more closely approximate the standard video sync signal frequencies. For those wishing to use an external video adapter to display the PET screen on a video monitor, this program can save having to modify the monitor.

The PET screen remains completely readable, with only minor narrowing of the picture and the possibility of loosing part of the bottom line in text mode. However, this can be corrected with a slight adjustment* to the PET video section.

Video adapters formerly used with 9" screen machines will not work directly with the new 12" machines as the polarity of the video out and horizontal sync signals have been reversed at the User Port. Correct this with:

```
POKE 59520,   12  :  POKE 59521,  0
```

This will give the desired effect on the external monitor but the PET screen will be inverted (but still readable). To get back to normal, POKE 59520, 12 : POKE 59521, 16

Some video interfaces for 9" machines have a horizontal sync position control and thus may accept the inverted pulses. People have used this system and it saved them having to obtain new interfaces.

Video adapters for 12" screen machines are now becoming more readily available and may be used with this program to produce an external picture which might otherwise be unable to "sync" to the strange signal frequencies coming from a 12" machine without this program.

```
100 DATA 0, 59, 2, 47, 4, 26, 5, 8, 7, 25, 9, 9
110 DATA 0, 59, 2, 47, 4, 33, 5, 6, 7, 30, 9, 7
120 FOR J=1 TO 6 : READ A, B
130 POKE 59520, A
140 POKE 59521, B
150 NEXT J
160 POKE 59468, 14 : PRINT CHR$(14)
170 PRINT "HIT 'STOP' FOR TEXT MODE CONFIGURATION"
180 GET A$ : IF A$ = " " THEN 90
190 FOR J=1 TO 6 : READ A, B
200 POKE 59520, A
210 POKE 59521, B
220 NEXT J
230 POKE 59468, 12 : PRINT CHR$(142)
240 PRINT "HIT 'STOP' FOR GRAPHICS MODE CONFIGURATION"
250 GET A$ : IF A$ = " " THEN 250
260 RESTORE : GOTO 120
```

*Refer to qualified service personnel

Programming Tips

Some Useful Hints on Dates

Have you ever wondered how nice it would be if the computer could tell you the date of the day 21 days from a given date? Whether you have or not, read on as there are several things that a computer can do with dates: not just store them!

Dates are one of the most awkward items to handle in computing, especially in languages like Basic, Comal, Pascal etc, which do not have user definable operators. For a start, there are several formats. For instance, Christmas 1982 could be written as

25 DEC 82
DEC 25, 1982
12/25/82
25/12/82

For the purposes of ease of writing, the last format will be used. There are several things that can be done with dates.

The ones covered here are

- Sorting in chronological order
- Conversion of a date to a number
- Conversion of a number to a date
- Input of a date
- Computing the day of the week
- Computing future or past dates

Sorting dates

The simplest manipulative operation on dates is a chronological sort. Dates can be easily sorted by reversing the positions of the day, month and year. This is given in ''sort' dates'' (lines 2000 to 2460).

Converting a date to a number

For most other operations on dates, two functions are required: one to convert a date to a number and another to convert that number back to a date. There are several formulae for doing the former but not all of them are reversible. The one given below is from the HP 25C Applications Programs Manual and is reversible. The algorithm is valid from March 1, 1900 to February 28, 2100 but the procedure given below is only valid from March 1, 1900 to December 31, 1999. This is given in ''days' val'' (lines 8000 to 8150).

Converting a number to a date

This procedure is semi-iterative (two iterations at most) to counter the truncation effects of the INT function. This is given in ''date'' (lines 9000 to 9300).

Input of a date

Date input can cause problems: especially with invalid dates. The problems like checking the number of days in each month or whether the date is that of a leap year are quite common. With the routines DATE and DAYS'VAL, the checking process can be simplified. This is given in ''input'date'' (lines 100 to 240).

Computing the day of the week

This is a very common use of dates in the commercial world simply because most people do not work on certain days of the week. The day of the week can be computed by taking the remainder of the result ofVAL after dividing it by 7. 0 means Sunday, 1 Monday .. 6 Saturday. This is given in 'weekday' (lines 1000 to 1040).

Computing past or future dates

Say you wish to tell a customer to come back in 21 days and it is 17/12/81 today and you do not have a calendar and he wants to know the actual date. No problem: just convert the date to a number, add 21 and convert the result back to a date. This is given in 'compute' (lines 4000 to 4160).

And to end ...

I hope the above routines which have been written in Comal (but can easily be translated into any other language) will open up new areas in programming for readers.

To end, here is a program to print a calendar. It is a Basic 2 program which was written on a 32K PET but might just fit on to a 16K PET. The date routine (41000) used here is the one given in A.J. Newey's ''One Hundred Computer Programming Problems'' (Pitman Press, 1973, ISBN 0 273 003080 9). The double SGN function is something left over from my Algol 60 days: it was faster and generated less code than an IF statement. On line 30050, if 29/02 is the same day of the week as 01/03 then it is not a leap year. It was written in a Fortran type style with all the 'declarations' on the top and subroutines at the bottom. This is not the most efficient format for programs on the PET but I did not know any better in those days.

Programming Tips

Listing of program 1.

```
0100 proc input'date(ref dmy$) closed
0120    dim temp$ of 8
0140    repeat
0160       input dmy$ // Get the date
0180       days:=days'val(dmy$)
0190       // Compute what it should be
0200       exec date(days,temp$)
0210       // Terminate when it is the same
0220    until dmy$=temp$
0240 endproc input'date

1000 proc weekday(dmy$) closed
1020    weekday:=days'val(dmy$) mod 7
1040 endproc weekday

2000 proc sort(ref dt$(),num'dates) closed
2010    dim temp$ of 8
2020    // Reverse dates
2040    for i:=1 to num'dates do
2060       dt$(i)=dt$(i,7:2)+dt$(i,4:2)+dt$(i,1:2)
2080    next i
2100    // Selection sort
2120    for i:=1 to num'dates-1 do
2140       min:=i
2160       for j:=i+1 to num'dates do
2180          if dt$(min)>dt$(j) then min:=j
2200       next j
2220       if i<>min then
2240          // Exchange
2260          temp$:=dt$(i);
2280          dt$(min):=dt$(i)
2300          dt$(i):=temp$
2320       endif
2340    next i
2360    // Put dates back in original form
2380    for i:=1 to num'dates do
2400       dt$(i):=dt$(i,5:2)+"/"
2420       dt$(i):+dt$(i,3:2)+"/"+dt$(i,1:2)
2440    next i
2460 endproc sort

4000 // compute future dates
4020 proc compute closed
4040    dim today$ of 8, ready$ of 8
4060    print "Today's date:",
4080    exec input'date(today$)
4100    input "Number of days:": wait'time
4120    exec date(days'val(today$)+wait'time,ready$)
4140    print "Come back on ",ready$
4160 endproc compute
```

```
8000 // convert a date to a number of days
8010 proc days'val(date$) closed
8020    round:=1/256
8030    // numval converts a string to a number
8040    day:=numval(date$(1:2))
8050    month:=numval(date$(4:2))
8060    year:=numval(date$(7:2))+1900
8070    if month>2 then
8080       month:+1
8090    else
8100       month:+13
8110       year:-1
8120    endif
8130    days'val:=day+int(month*30.6+round)
8140    days'val:+int(year*365.25+round)
8150 endproc days'val

9000 // convert a number of days to a date
9010 proc date(days,ref res$) closed
9020    round:=1/256
9030    year:=int(days/365.25+round)
9040    // evaluate the month
9050    repeat
9060       year2:=int(year*365.25+round)
9070       month:=int((days-year2)/30.6+round)
9080       if month<4 then year:-1
9090    until month>=4
9100   // evaluate the day
9110    repeat
9120       day:=days-int(month*30.6+round)-year2
9130       if day=0 then month:-1
9140    until day>0
9150    // convert to a string
9160    res$:=""
9170    if month>13 then
9180       year:+1
9190       month:-13
9200    else
9210       month:-1
9220    endif
9230    // the routine str does the following:
9240    // res$=res$+right$(str$(x+100),2)
9250    exec str(day,res$,2)
9260    date$:=date$+"/"
9270    exec str(month,res$,2)
9280    date$:=date$+"/"
9290    exec str(year-1900,res$,2)
9300 endproc date
```

```
10000  REM==CALENDAR  (5 LINE VERSION)
10020  REM  WRITTEN: 16/03/80, CUP
10030  REM  UPDATED: 08/06/82, CUP
10053  REM ---------------------------
10055  REM THE AUTHOR IS NOT RESPONSIBLE
10056  REM FOR ANY  INCORRECT  CALENDARS
10057  REM GENERATED  BY  THIS  PROGRAM.
10058  REM ---------------------------

10060  REM==PROGRAM  ALTERATIONS
10062  REM  10123&15730: WIDE PRINT (W$)
10063  REM  10114&15730: NORMAL PRINT (N$)
10064  REM  11000-11999: INTRODUCTION
10065  REM  10119,20700: TRAILER
10066  REM  10630,10690,36010: DIFFERENT PRINT INSTRUCTIONS
10068  REM  50000-53500: LARGE CHARACTERS
10070  REM  54000      : MNEMONICS FOR DAYS
10072  REM  55000-55110: SPELLING OF MONTHS
10074  REM  56000-56180: MICRO OTHER THAN 6502

10100  REM==VARIABLES
10101  A$="":
       A=ASC("a")
10102  B$=" ":
       B=144
10103  C=0:
       C$="":
       CO=0
10104  D=0:
       DY=0
10108  HI=0
10112  L=0:
       L$="":
       LC=65:
       LN=0:
       LO=0
10113  M=0:
       MO=0:
       M1=0:
       M2=0:
       M$=""

10114  REM--SET UP FOR BD80P
       N=0:
       NC=0:
       ND=0:
       N$=CHR$(27)+CHR$(14):
10116  P=0
10117  Q$=CHR$(34)
10119  S$="cup/pet
10120  TY=0
10122  V$=CHR$(124)

10123  REM--SET UP FOR BD80P
       W=0:
       W$=CHR$(27)+CHR$(15):
10125  Y=0:
       YO=0

10200  REM==ARRAYS
10212  DIM LT$(25)
10213  DIM MD(4,6),MN$(12),MN(12)
10214  DIM NM$(9)
10216  DIM P$(65)
10220  DIM T(13)
10223  DIM WK(6)

10500  REM==MAIN  PROGRAM

10505  REM--INTRODUCTION
       GOSUB 11000:

10510  REM--INITIALIZE
       GOSUB 20000:
10520  INPUT "{clear.screen}print symbol";A$:
       IF LEN(A$)=1 THEN 10550
10530  PRINT "{5cursor.down}{rvs.on}mmm...";:
       GOSUB 38000
10540  PRINT "i can't decide which one to use":
       GOSUB 38000:
       GOTO 10520

10550  REM--TEXT ETC
       GOSUB 42000:
       GOSUB 15000:
10560  PRINT "{clear.screen}year        ";:
       LO=1900:
       HI=9999:
       GOSUB 40000:
       Y=N
10570  PRINT "starting month";:
       LO=1:
       HI=12:
       GOSUB 40000:
       M1=N
10580  PRINT "ending month  ";:
       LO=N:
       HI=12:
       GOSUB 40000:
       M2=N
10590   INPUT "ordinary cal  ";A$:
       TY=-(A$="y")
10600  INPUT "diary type     ";A$:
       TY=TY-(A$="y")*2
10605  INPUT "number of cals";NC:
       IF NC<1 THEN 10605
10610  IF TY=0 THEN 10590
10620  FOR C=1 TO NC
10630     OPEN 222,4:
          GOSUB 16000
10640     FOR M=M1 TO M2

10650        REM--FORM THE MONTH
             GOSUB 30000:
10660        IF TY AND 1 THEN
```

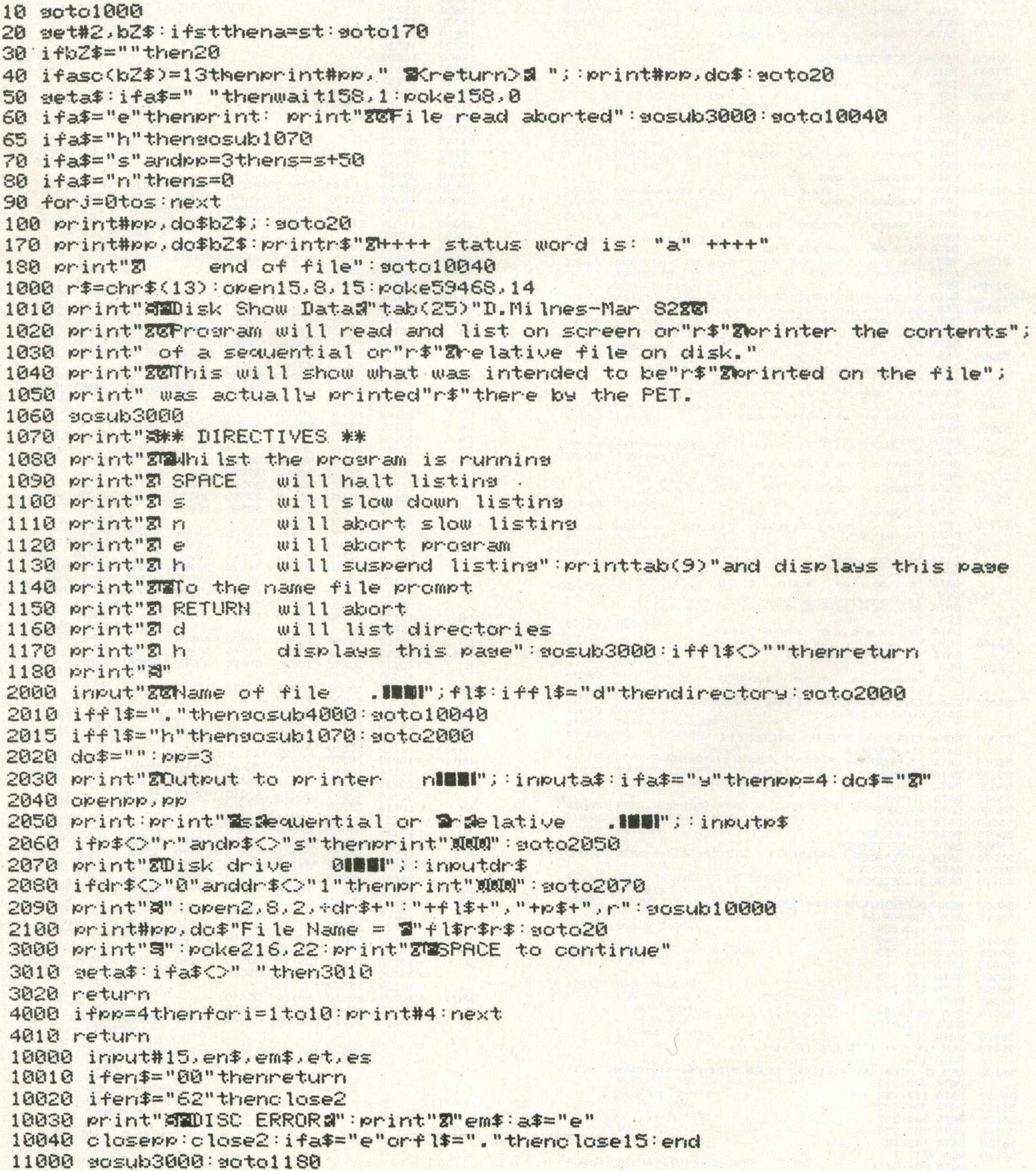
```
           GOSUB 35000
10670      IF TY AND 2 THEN
           GOSUB 34000
10680    NEXT M
10690    CLOSE 222
10700  NEXT C
10710  GOTO 10520

11000  REM==INTRODUCTION
11003  PRINT "{clear.screen}{8cursor.down}
         -----------------------------
11005  PRINT "      the author is not responsible
11006  PRINT ".     for any  incorrect  calendars
11007  PRINT "      generated  by  this  program.
11008  PRINT "      -----------------------------
11009  GOSUB 37000
11010  PRINT "{clear.screen}this program will generate a calendar
11020  PRINT "on the printer for any year between
11030  PRINT "1900 and 9999 (if the system doesn't
11040  PRINT "change by then).  each month will be
11050  PRINT "printed on a page of 80 columns by
11055  PRINT "66 lines.
11060  PRINT "{cursor.down}the computer will first ask you for
11070  PRINT "'print symbol'.  this is the symbol
11080  PRINT "with which the 'large' characters will
11090  PRINT "be printed.  <return> terminates the
11095  PRINT "program.
11100  PRINT "{cursor.down}it will then print 'text': this allows
11105  PRINT "the entry of text before the calendar.
11110  PRINT "each line must begin with one of the
11120  PRINT "following codes:
11130  PRINT "{cursor.down}    b  blank line
11140  PRINT "    e  end of text
11150  PRINT "    l  large characters
11160  PRINT "    n  normal characters
11170  PRINT."    w  wide characters
11175  GOSUB 37000
11180  PRINT "{cursor.down}up to 66 lines of text are allowed.
11190  PRINT "this depends on the option chosen (l
11200  PRINT "takes 5 lines).  the maximum length of
11210  PRINT "each line also depends on the option
11220  PRINT "chosen (l-13, n-72, w-40).  you have
11230  PRINT "the choice of either centralizing the
11240  PRINT "output or having it 'as is'.
11250  PRINT "{cursor.down}the computer will then ask you for the
11260  PRINT "year.  this should be given as a four
11270  PRINT "digit number between 1900 and 9999.
11280  PRINT "{cursor.down}this is followed by a request for the
11290  PRINT "months for which the calendar is
11300  PRINT "required.  the input should be given
11310  PRINT "numerically: 1=jan, 2=feb, .. 12=dec.
11320  GOSUB 37000
11330  PRINT "you will then have the choice of an
11340  PRINT "ordinary calendar, a diary type or both
11360  PRINT "{cursor.down}finally, you are warned that continuous
11370  PRINT "printing of wide characters may
11380  PRINT "overheat your printer.
11390  PRINT "{2cursor.down}p.s. the only characters allowed in
          the
11400  PRINT "'large' option are alphanumerics and
11410  PRINT "spaces. blanks will be substitued for
11420  PRINT "'unknown' characters.
11430  GOSUB 37000
11999  RETURN

15000  REM==OBTAIN THE FRONT PAGE
15020  PRINT "{cursor.home}{3cursor.down}{rvs.on}text{rvs.off}":
       INPUT "{cursor.home}{5cursor.down}same as previous
          page";A$:
       A$=LEFT$(A$,1)
15030  IF A$="y" THEN
          RETURN
15040  IF A$<>"n" THEN 15020
15100  INPUT "{cursor.home}{7cursor.down}should the text be
          centralized";C$:
       C$=LEFT$(C$,1)
15110  IF C$<>"y" AND C$<>"n" THEN
          GOTO 15100
15200  PRINT "lines{cursor.down}{5cursor.left}left":
       LC=66
15210  LC=LC-1:
       IF LC<0 THEN
          LC=0:
          RETURN
15220  PRINT LC;TAB(4);"{3cursor.right}"Q$Q$"{cursor.left}
          {3cursor.left}";:
       INPUT L$:
       A$=LEFT$(L$,1)
15230  IF A$="b" THEN
          P$(LC)="":
          GOTO 15210
15240  IF A$="e" THEN
          RETURN
15250  L$=MID$(L$,2):
       LN=LEN(L$):
       IF A$="l" THEN 15500
15260  IF A$="n" THEN 15600
15270  IF A$="w" THEN 15700
15280  PRINT "{rvs.on}unknown character code{rvs.off}":
       GOTO 15220

15500  REM--LARGE CHARACTERS
15510  IF LN>13 THEN 15810
15520  IF LC<4 THEN 15820
15530  FOR P=1 TO 21 STEP 5
15540     A$="":
          IF C$="y" THEN
             A$=LEFT$(B$,40-LN*3)
15550     FOR N=1 TO LN
15552        M$=MID$(L$,N,1)
15554        IF M$>="0" AND M$<="9" THEN
                A$=A$+MID$(NM$(VAL(M$)),P,5)+" ":
                GOTO 15560
15556        IF M$>="a" AND M$<="z" THEN
                A$=A$+MID$(LT$(ASC(M$)-A),P,5)+" ":
                GOTO 15560
15558        IF M$<>" " THEN
                PRINT "{rvs.on}unknown char: "+M$"{rvs.off}'
15559        A$=A$+LEFT$(B$,6)
15560     NEXT N
15570     P$(LC)=A$:
          IF P<>21 THEN
             LC=LC-1
15580  NEXT P
15590  GOTO 15210

15600  REM--NORMAL CHARACTERS
15610  IF LN>80 THEN 15810
15620  A$="":
       IF C$="y" THEN
          A$=LEFT$(B$,INT((80-LN)/2))
15630  P$(LC)=A$+L$:
       GOTO 15210

15700  REM--WIDE CHARACTERS
15710  IF LN>40 THEN 15810
15720  A$="":
       IF C$="y" THEN
          A$=LEFT$(B$,40-LN)
15730  P$(LC)=A$+W$+L$+N$:
       GOTO 15210

15800  REM--ERROR MESSAGES
15810  PRINT "{rvs.on}string too long{rvs.off}":
       GOTO 15220
15820  PRINT "{rvs.on}no more space on page{rvs.off}":
       GOTO 15220
16000  REM==PRINT FRONT PAGE
16020  IF LC=65 THEN
          RETURN
16100  L=INT(LC/2):
       IF LC<>0 THEN
          L$="":
          FOR I=1 TO L:
             GOSUB 36000:
          NEXT I
16110  FOR I=65 TO LC STEP -1:
          L=L+1:
          L$=P$(I):
          GOSUB 36000:
       NEXT I

16120  REM--FORM FEED
       IF L<>66 THEN
          L$="":
          FOR I=L TO 66:
             GOSUB 36000:
          NEXT I:
16999  RETURN

20000  REM==INITIALIZE

20100  REM--NUMBERS
20110  FOR N=0 TO 9:
          READ NM$(N):
          NM$(N)=NM$(N)+"":
       NEXT N

20200  REM--LETTERS
20210  FOR N=0 TO 25:
          READ LT$(N):
          LT$(N)=LT$(N)+"":
       NEXT N

20300  REM--WEEKDAYS
20310  FOR W=0 TO 6:
          READ L$:
          WK(W)=ASC(L$)-A:
       NEXT W

20400  REM--B$=128 BLANKS
20410  FOR N=1 TO 6:
          B$=B$+B$:
       NEXT N

20500  REM--MONTHS
20510  FOR N=1 TO 12:
          READ MN(N),MN$(N):
       NEXT N

20600  REM--CHARACTER CONVERSION ROUTINE
20610  REM  CHECKSUM
       CS=0:
20620  FOR N=826 TO 1024:
          READ CO:
          CS=CS+CO:
          IF CO>0 THEN
             POKE N,CO:
          NEXT N
20630  IF CS<>0 THEN
          PRINT "{rvs.on}Error in Assembler program{rvs.off}":
          LIST 56000-56180

20660  REM--IRQ
       IF PEEK(40)+PEEK(41)*256<>1025 THEN
          B=537:

20700  REM--PUT TRAILER IN BOTTOM RIGHT CORNER
20710  S$=RIGHT$(B$+S$,80)
20999  RETURN

30000  REM==FORM THE MONTH

30010  REM--DETERMINE THE NUMBER OF DAYS
30020  ND=MN(M)
30030  IF M<>2 THEN 30100

30040  REM--SPECIAL CASE FOR FEBRUARY
30050  D=1:
       M=3:
        GOSUB 41000:
        N=DY:
        D=29:
        M=2:
        GOSUB 41000:
        IF DY=N THEN
           ND=28

30100   REM--CLEAR MONTH
30110   FOR W=0 TO 4:
           FOR DY=0 TO 6:
              MD(W,DY)=0:
           NEXT DY:
        NEXT W

30200   REM--FILL IN THE MONTH
30210   D=1:
        GOSUB 41000:
        W=0
30220   FOR D=1 TO ND
30240      MD(W,DY)=D:
           DY=DY+1:
           IF DY=7 THEN
              DY=0:
              W=W+1:
              IF W=5 THEN
                 W=0
30290   NEXT D
30300   RETURN

31000   REM==OUTPUT THE MONTH HEADER

31010   REM--FILL IN THE TITLE (MONTH)
31020   M$=MN$(M):
        L=LEN(M$)
31030   FOR N=1 TO L:
           T(N)=ASC(MID$(M$,N,1))-A:
        NEXT N

31100   REM--FILL IN THE YEAR
31110   YO=Y
31120   FOR N=13 TO 10 STEP -1:
           YO=INT(YO)/10:
           T(N)=INT((YO-INT(YO))*10+0.5):
        NEXT N

31200   REM--OUTPUT THE MONTH AND YEAR
31220   FOR P=1 TO 21 STEP 5
31240      L$="":
           FOR N=1 TO L:
              L$=L$+MID$(LT$(T(N)),P,5)+" ":
           NEXT N
31310      L$=LEFT$(L$+B$,56):
           FOR N=10 TO 13:
              L$=L$+" "+MID$(NM$(T(N)),P,5):
           NEXT N
31340      GOSUB 36000
31350   NEXT P
31360   L$="":
        GOSUB 36000
31999   RETURN

32000   REM==DIARY SEPARATOR
32010   L$=LEFT$(B$,7)+"+"
32020   FOR W=0 TO 4:
           L$=L$+"-------------+":
        NEXT W
32030   GOSUB 36000
32999   RETURN

34000   REM==DIARY
34010   GOSUB 31000:
        L$="":
        GOSUB 36000:
        GOSUB 32000
34100   FOR DY=0 TO 6
34110      FOR P=-4 TO 26 STEP 5

34120         REM--OBTAIN THE DAY OF THE WEEK
34125         IF P=-4 OR P=26 THEN
                 L$=LEFT$(B$,7)+V$:
                 GOTO 34140
34130         L$=MID$(LT$(WK(DY)),P,5)+"  "+V$

34140         REM--DATES WHICH FALL ON THAT DAY
34150         FOR W=0 TO 4
34160            IF P=-4 AND MD(W,DY)<>0 THEN
                    L$=L$+RIGHT$(B$+STR$(MD(W,DY)),13)+V$:
                    GOTO 34180
34170            L$=L$+RIGHT$(B$,13)+V$
34180         NEXT W
34190         GOSUB 36000
34200      NEXT P
34210      GOSUB 32000
34220   NEXT DY

34999   REM--TRAILER
        L$=S$:
        GOSUB 36000:
        L$="":
        GOSUB 36000:
        RETURN:

35000   REM==NORMAL CALENDAR
35010   GOSUB 31000:
        L$="":
        GOSUB 36000:
        GOSUB 36000:
        GOSUB 36000

35400   REM--OUTPUT THE DAYS OF THE MONTH
35410   FOR DY=0 TO 6
35420      FOR P=1 TO 21 STEP 5

35430         REM--OBTAIN THE DAY OF THE WEEK
35440         L$=MID$(LT$(WK(DY)),P,5)

35450         REM--DATES WHICH FALL ON THAT DAY
35460         FOR W=0 TO 4
35470            D=MD(W,DY):
                 L$=L$+"    "
35480            IF D<10 THEN
                    L$=L$+"      ":
                    GOTO 35500
35490            L$=L$+MID$(NM$(INT(D/10)),P,5)+" "
35500            IF D=0 THEN
                    L$=L$+"  ":
                    GOTO 35520
35510            L$=L$+MID$(NM$(D-INT(D/10)*10),P,5)
35520         NEXT W
35530         GOSUB 36000
35540      NEXT P

35550      REM--TRAILER
           L$="":
           GOSUB 36000:
           GOSUB 36000:
           IF DY=6 THEN
              L$=S$:
35560      GOSUB 36000
35570   NEXT DY
35580   L$="":
        GOSUB 36000:
        RETURN

36000   REM==PRINT A LINE
36010   PRINT £222,L$
36020   RETURN

37000   REM==WAIT

37010   REM--IN CASE SOME IDIOT DECIDES TO USE RUN/STOP
        P=PEEK(B):
        POKE B,P+3:
37020   PRINT "{cursor.home}{24cursor.down}{rvs.on}{6cursor.right}
           strike any key to continue{rvs.off}{2cursor.up}
37030   GET A$:
        IF A$="" THEN 37030
37040   POKE B,P
37050   IF (ASC(A$) AND 127)=3 THEN
           PRINT "{5cursor.down}trust you to do
              that!{5cursor.down}":
           STOP
37060   PRINT "{clear.screen}"
37999   RETURN

38000   REM==WAIT FOR A WHILE
38010   FOR P=1 TO 500:
           GET A$:
           IF A$="" THEN
           NEXT P
38020   RETURN

40000   REM==INPUT ROUTINE
40010   INPUT N
40020   IF N<LO OR N>HI THEN
           PRINT "{rvs.on}should be between"LO;" and"HI;
           GOTO 40000
40030   IF N<>INT(N) THEN
           PRINT "{rvs.on}should be an integer{rvs.off}";
           GOTO 40000
40999   RETURN

41000   REM==DETERMINE THE DAY OF THE
        WEEK
41010   REM--0=SUN .. 6=SAT
41020   CO=SGN(SGN(M-2)-1)
41030   MO=M-10*CO-2*(CO+1)
41040   YO=Y+CO
41050   CO=INT(YO/100)
41060   YO=YO-CO*100

41065   REM--WARNING: POSSIBLE ROUNDING ERRORS
41070   DY=INT((26*MO-2)/10)+D+YO+INT(YO/4)+INT(CO/4)-2*CO
41080   DY=DY-196*(SGN(SGN(DY)+1)-1)

41085   REM--WARNING: POSSIBLE ROUNDING ERRORS
41090   DY=INT(DY-INT(DY/7)*7+0.5)
41100   RETURN

42000   REM==SYMBOL CONVERSION

42010   REM--TRYING TO BE CLEVER?
42020   IF A$<>" " THEN 42100
42030   PRINT "{4cursor.down}you have the author's congrats for
              program crashing!!!"
42040   GOSUB 38000

42100   REM--THE NEW SYMBOL
        POKE 254,ASC(A$):

42110   REM--NUMBERS
        FOR N=0 TO 9:
           A$=NM$(N):
           SYS 826:
           NM$(N)=A$:
        NEXT N:

42120   REM--LETTERS
        FOR N=0 TO 25:
           A$=LT$(N):
           SYS 826:
           LT$(N)=A$:
        NEXT N:
42999   RETURN

50000   REM=NUMBERS
50001   REM  0
        DATA " ooo o   oo   oo   o ooo ":
```

```
50100  REM  1
       DATA "  o   oo    o     o    ooo ":
50200  REM  2
       DATA " ooo o   o  oo   o    ooooo":
50300  REM  3
       DATA " ooo  .   o  oo       o ooo ":
50400  REM  4
       DATA "    o   oo   o o ooooo    o ":
50500  REM  5
       DATA "oooooo      oooo       ooooo ":
50600  REM  6
       DATA " ooo o      oooo o    o ooo ":
50700  REM  7
       DATA "ooooo      o    o    o    o   ":
50800  REM  8
       DATA " ooo o   o ooo o    o ooo ":
50900  REM  9
       DATA " ooo o   o oooo      o ooo ":

51000  REM=LETTERS
51001  REM  A
       DATA " ooo o   ooooooo   oo    o":
51100  REM  B
       DATA "oooo o   ooooo o   ooooo ":
51200  REM  C
       DATA " ooooo     o     o      oooo":
51300  REM  D
       DATA "oooo o   oo    oo    ooooo ":
51400  REM  E
       DATA "oooooo      ooo   o      ooooo":
51500  REM  F
       DATA "oooooo      ooo   o      o    ":
51600  REM  G
       DATA " ooooo      o  ooo   o ooo ":
51700  REM  H
       DATA "o    oo   ooooooo   oo    o":
51800  REM  I
       DATA " ooo    o      o      o    ooo ":
51900  REM  J
       DATA "    o     o     o o   o   oo   ":
52000  REM  K
       DATA "o    oo  o ooo   o   o o    o":
52100  REM  L
       DATA "o     o      o      o      ooooo":
52200  REM  M
       DATA "o   ooo ooo o oo    oo    o":
52300  REM  N
       DATA "o   ooo   oo o oo   ooo    o":
52400  REM  O
       DATA " ooo o    oo    oo    o ooo ":
52500  REM  P
       DATA "oooo o   ooooo o     o      ":
52600  REM  Q
       DATA " ooo o    oo o oo   o   oo o":
52700  REM  R
       DATA "oooo o   ooooo o    oo    o":
52800  REM  S
       DATA " ooo o        ooo       o ooo ":
52900  REM  T
       DATA "ooooo   o      o      o      o  ":
53000  REM  U
       DATA "o    oo    oo    oo  . o ooo ":
53100  REM  V
       DATA "o    oo    oo    o o o    o   ":
53200  REM  W
       DATA "o    oo    oo o oo o o o o ":
53300  REM  X
       DATA "o    o o o    o    o o o    o":
53400  REM  Y
       DATA "o    oo    o ooo    o      o   ":
53500  REM  Z
       DATA "ooooo    o     o     o    ooooo":

54000  REM=DAYS  OF  THE  WEEK
       DATA S,M,T,W,T,F,S:

55000  REM=DAYS  PER  MONTH
       DATA 31,JANUARY:
55010  DATA 28,FEBRUARY
55020  DATA 31,MARCH
55030  DATA 30,APRIL
55040  DATA 31,MAY
55050  DATA 30,JUNE
55060  DATA 31,JULY
55070  DATA 31,AUGUST
55080  DATA 30,SEPTEMBER
55090  DATA 31,OCTOBER
55100  DATA 30,NOVEMBER
55110  DATA 31,DECEMBER

56000  REM==CONVERT  CHARS
56005  REM PTR=ADR(A$)
       DATA 160,004:
56010  DATA 177,042
56020  DATA 133,253
56030  DATA 136
56040  DATA 177,042
56050  DATA 133,252
56060  DATA 136
56070  REM Y=LEN(A$)
       DATA 177,042:
56080  DATA 168
56090  REM FOR Y=Y-1 TO 0 STEP -1
       DATA 136:
56100  REM IF MID$(A$,Y+1,1)<>" "THEN MID$(A$,Y+1,1)=CH$
       DATA 048,012:
56110  DATA 177,252
56120  DATA 201,032
56130  DATA 240,247
56140  DATA 165,254
56150  DATA 145,252
56160  REM NEXT Y
       DATA 208,241:
56170  REM RETURN
       DATA 096:
```

```
56180  DATA -4738
63998  REM   SCRATCH"calendar",D0:DSAVE"calendar",D0:VERIFY"*",8
63999  REM   SCRATCH"calendar",D1:DSAVE"calendar",D1:VERIFY"*",8
10520   10540  10710
10550   10520
10590   10610
10605   10605
11000   10505
15000   10550
15020   15040
15100   15110
15210   15230  15590  15630  15730
15220   15280  15810  15820
15500   15250
15560   15554  15556
15600   15260
15700   15270
15810   15510  15610  15710
15820   15520
16000   10630
20000   10510
30000   10650
30100   30030
31000   34010  35010
32000   34010  34210
34000   10670
34140   34125
34180   34160
35000   10660
35500   35480
35520   35500
36000   16100  16110  16120  31340  31360  32030  34010  34190
        34999  35010  35530  35550  35560  35580
37000   11009  11175  11320  11430
37030   37030
38000   10530  10540  42040
40000   10560  10570  10580  40020  40030
41000   30050  30210
42000   10550
42100   42020
56000   20630
A       10101  15556  20310  31030
A$      10101  10520  10590  10600  15020  15030  15040  15220
        15230  15240  15250  15260  15270  15540  15554  15556
        15559  15570  15620  15630  15720  15730  37030  37050
        38010  42020  42100  42110  42120
B       10102  20660  37010  37040
B$      10102  15540  15559  15620  15720  20410  20710  31310
        32010  34125  34160  34170
C       10103  10620  10700
C$      10103  15100  15110  15540  15620  15720
CO      10103  20620  41020  41030  41040  41050  41060  41070
CS      20610  20620  20630
D       10104  30050  30210  30220  30240  30290  35470  35480
        35490  35500  35510  41070
DY      10104  30050  30110  30240  34100  34130  34160  34220
        35410  35440  35470  35550  35570  41070  41080  41090
HI      10108  10560  10570  10580  40020
I       16100  16110  16120
L       10112  16100  16110  16120  31020  31030  31240
L$      10112  15220  15250  15552  15630  15730  16100  16110
        16120  20310  31240  31310  31360  32010  32020  34010
        34125  34130  34160  34170  34999  35010  35440  35470
        35480  35490  35500  35510  35550  35580  36010
LC      10112  15200  15210  15220  15230  15520  15570  15630
        15730  16020  16100  16110
LN      10112  15250  15510  15540  15550  15610  15620  15710
        15720
LO      10112  10560  10570  10580  40020
LT$()   10212  15556  20210  31240  34130  35440  42120
M       10113  10640  10680  30020  30030  30050  31020  41020
        41030
M$      10113  15552  15554  15556  15558  31020  31030
MO      10113  41030  41070
M1      10113  10570  10640
M2      10113  10580  10640
MD()    10213  30110  30240  34160  35470
MN$()   10213  20510  31020
MN()    10213  20510  30020
N       10114  10560  10570  10580  15550  15552  15560  20110
        20210  20410  20510  20620  30050  31030  31120  31240
        31310  40010  40020  40030  42110  42120
N$      10114  15730
NC      10114  10605  10620
ND      10114  30020  30050  30220
NM$()   10214  15554  20110  31310  35490  35510  42110
P       10116  15530  15554  15556  15570  15580  31220  31240
        31310  31350  34110  34125  34130  34160  34200  35420
        35440  35490  35510  35540  37010  37040  38010
P$()    10216  15230  15570  15630  15730  16110
Q$      10117  15220
S$      10119  20710  34999  35550
T()     10220  31030  31120  31240  31310
TY      10120  10590  10600  10610  10660  10670
V$      10122  34125  34130  34160  34170
W       10123  20310  30110  30210  30240  32020  34150  34160
        34180  35460  35470  35520
W$      10123  15730
WK()    10223  20310  34130  35440
Y       10125  10560  31110  41040
YO      10125  31110  31120  41040  41050  41060  41070
```

Programming Tips

Disk Show Data



```
10 goto1000
20 get#2,bZ$:ifstthena=st:goto170
30 ifbZ$=""then20
40 ifasc(bZ$)=13thenprint#pp," ▒<return>▒ ";:print#pp,do$:goto20
50 geta$:ifa$=" "thenwait158,1:poke158,0
60 ifa$="e"thenprint: print"▒▒File read aborted":gosub3000:goto10040
65 ifa$="h"thengosub1070
70 ifa$="s"andpp=3thens=s+50
80 ifa$="n"thens=0
90 forj=0tos:next
100 print#pp,do$bZ$;:goto20
170 print#pp,do$bZ$:printr$"▒++++ status word is: "a" ++++"
180 print"▒     end of file":goto10040
1000 r$=chr$(13):open15,8,15:poke59468,14
1010 print"▒▒Disk Show Data▒"tab(25)"D.Milnes-Mar 82▒▒
1020 print"▒▒Program will read and list on screen or"r$"▒printer the contents";
1030 print" of a sequential or"r$"▒relative file on disk."
1040 print"▒▒This will show what was intended to be"r$"▒printed on the file";
1050 print" was actually printed"r$"there by the PET.
1060 gosub3000
1070 print"▒** DIRECTIVES **
1080 print"▒▒Whilst the program is running
1090 print"▒ SPACE    will halt listing .
1100 print"▒ s        will slow down listing
1110 print"▒ n        will abort slow listing
1120 print"▒ e        will abort program
1130 print"▒ h        will suspend listing":printtab(9)"and displays this page
1140 print"▒▒To the name file prompt
1150 print"▒ RETURN   will abort
1160 print"▒ d        will list directories
1170 print"▒ h        displays this page":gosub3000:iffl$<>""thenreturn
1180 print"▒"
2000 input"▒▒Name of file   .███";fl$:iffl$="d"thendirectory:goto2000
2010 iffl$="."thengosub4000:goto10040
2015 iffl$="h"thengosub1070:goto2000
2020 do$="":pp=3
2030 print"▒Output to printer   n███";:inputa$:ifa$="y"thenpp=4:do$="▒"
2040 openpp,pp
2050 print:print"▒s▒equential or ▒r▒elative   .███";:inputp$
2060 ifp$<>"r"andp$<>"s"thenprint"▒▒▒":goto2050
2070 print"▒Disk drive   0███";:inputdr$
2080 ifdr$<>"0"anddr$<>"1"thenprint"▒▒▒":goto2070
2090 print"▒":open2,8,2,+dr$+":"+fl$+","+p$+",r":gosub10000
2100 print#pp,do$"File Name = ▒"fl$r$r$:goto20
3000 print"▒":poke216,22:print"▒▒SPACE to continue"
3010 geta$:ifa$<>" "then3010
3020 return
4000 ifpp=4thenfori=1to10:print#4:next
4010 return
10000 input#15,en$,em$,et,es
10010 ifen$="00"thenreturn
10020 ifen$="62"thenclose2
10030 print"▒▒DISC ERROR▒":print"▒"em$:a$="e"
10040 closepp:close2:ifa$="e"orfl$="."thenclose15:end
11000 gosub3000:goto1180
63999 scratch"disk show data":dsave"disk show data"
```

commodore COMPUTER

PET PACK

BRING TO YOU, NOW ON CASSETTE AND DISK GAMES OF THRILLS & SKILL FOR ALL THE FAMILY

NEW PROGRAMS

Best of Arcade brings together the three most popular Petpack games, Invaders, Cosmic Jailbreak and Cosmiads. These old favourites have been updated to run on 80-column machines also!

MPD 121 BEST OF ARCADE £22.50 - DISK PACK

Best of Treasure Trove gives you twenty of the best games from the Treasure Trove series, including four arcade-type games, Night Drive, Car Race, Breakout and Money Table! There are simulation games, brainteasers and more, making this package the best value ever in games!

MPD 122 BEST OF TREASURE TROVE £22.50 - DISK PACK

Assembler Tutorial is an extremely well thought out cassette-based package which teaches Assembly Language programming. Now for the first time, you can sit at your computer and learn at your own speed with this self contained course combining lessons with hands-on practice!

MP 124 ASSEMBLER TUTORIAL £50.00

Resident Assembler for all PETs
With excellent documentation and examples.

MP119 RAMP £22.50

Disk Packs available in either 8050 - D8 format or 3040/4040 - D4 format. Please state D8 or D4 when ordering.
Prices include VAT and P&P.

PUB GAMES

This latest disk package brings you five totally new games, never before seen on a PET screen! The programs will all run on 80-column machines also!

DISASTEROIDS - Your mission - pilot your spaceship through the treacherous asteroid belt using your lasers to blast as many asteroids as possible. The PET version of the famous arcade game!

STELLAR WARS - Your spaceship is being pursued by the fighter ships of the evil Empire. You must take control of the ship's laser cannon. Get the fighters in your sights and blast away. The future of the universe depends on your skill and accuracy.

WAR! - You are the captain of the British torpedo boat. You must steer your ship through the minefield to destroy the four shore bases. Unfortunately, you only have two torpedos at a time, so you must get through the minefield again to rearm. Prove that we still rule the waves!

STAR FIGHT - The Alien Invaders are coming again! Control your laser cannon to blast their ships out of the sky and destroy their missiles and bombs. Accuracy and quick reflexes are essential.

DEPTH CHARGE - Your mission - seek out and destroy the enemy submarines. They are armed with mines which they release to float up at you. You must control the position of your ship and fire your depth charges to destroy as many subs and mines as possible without being hit.

MPD 123 PUB GAMES £22.50 - DISK PACK

These games are available separately on Cassette at £6.99 each!

**MANUFACTURED AND DISTRIBUTED BY AUDIOGENIC LTD.
AVAILABLE FROM GOOD DEALERS, OR DIRECT FROM:**

**AUDIOGENIC, P.O. BOX 88, READING, BERKS.
Tel: (0734) 586334**

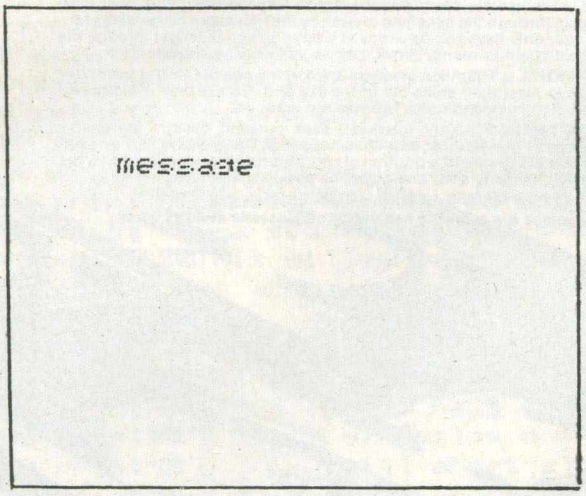
Basic Programs

Cursor Positioning on the Pet

One of the most popular (?) and easiest ways to position the cursor in a particular position on the PET screen and displaying a message at that position is as follows:-

```
1.      10 PRINT"◘◘◘◘◘◘◘◘◘◘◘◘"TAB(12)
"MESSAGE"
```

which will appear as shown below:-



message

Another method which I favoured for a long time was to initialise two strings —cr$(CURSOR RIGHT) and cd$(CURSOR DOWN) to 39 cursor rights and 24 cursor downs respectively.

```
2.      10 FORI=1TO6:CR$=CR$+"▶▶▶▶▶▶"
:NEXT
        20 CR$=CR$+"▶▶▶
        30 CD$=CD$+"◘◘◘◘◘◘◘◘◘◘◘◘◘◘◘◘◘
◘◘◘◘◘◘◘"
        40 PRINT LEFT$(CD$,12)LEFT$
(CR$,8)"MESSAGE"
```

which would do the same as number 1.

The positioning of the message on the screen could be changed by varying the numbers 12 and 8 in left$(cd$,12) and left$(cr$,8).

3. The third way is to use the memory locations which hold the row position and column position of the cursor — 216 and 198 respectively.

```
Thus    10 PRINT"■":POKE216,12:POKE
198,8:PRINT"MESSAGE
```

would give similar results as the previous example.

Both the last two examples could be incorporated in a sub-routine as follows:-

```
10 CR$= < 39 CURSOR RIGHTS >

20 CD$= < 24 CURSOR DOWNS >

30 A$="MESSAGE":R=12:D=8

40 GOSUB 2000

50 ..........

60 ..........



2000 PRINT LEFT$(CD$,D)LEFT$(CR$,R)
A$":RETURN

 and similarly for the poke example.
```

One other method is to use a short machine code program which uses the built in ROM routines of the PET.

a. To evaluate the position of the cursor

b. Position the cursor on the screen and then printing the string.

I use this routine now and extend it to blank out portions of the screen as an extra facility. The routine is relocatable and in fact I tuck it behind a REM statement at the beginning of the program. More about that later.

The SYNTAX of the routine is

sys(n),r,c,1:Printas

where n = entry point of machine code
r = row which cursor is to be positioned
c = column where cursor is to be positioned
l = 0 (zero) for the print facility or any value less than 256 for the blank function.

The routine is shown as assembled in the second cassette buffer (sys826 to call)

a) as a HEX dump
b) in assembler for those with this facility

```
100  033A              ; ********************************
120  033A              ; * D.MILNES..  FEBRUARY 1982
130  033A              ; * ----------------------------
140  033A              ; * PRINTS STRING AT
150  033A              ; * POSITION ON
160  033A              ; * SCREEN
170  033A              ; *
180  033A              ; * SYNTAX.SYS(XXXX),R,C,L:PRINTA$
190  033A              ; * ----------------------------
210  033A              ; * BLANKS PORTION
220  033A              ; * OF SCREEN
230  033A              ; *
240  033A              ; * SYNTAX....SYS(XXXX),R,C,L
250  033A              ; * ----------------------------
260  033A              ; *
270  033A              ; * R=ROW C=COLUMN L=LENGTH
280  033A              ; *
282  033A              ; * ++ L=0 FOR STRING PRINT ++
283  033A              ; *
284  033A              ; * ++ RELOCATABLE - BYTES=47 ++
290  033A              ; ******************************
300  033A              ;
310  033A              ;
500  033A                      *=826
510  033A  48                          PHA
520  033B  8A                          TXA
530  033C  48                          PHA
540  033D  98                          TYA
550  033E  48                          PHA
560  033F  20F5BE                      JSR   $BEF5              ;CHECKS FOR COMMA
570  0342  20D4C8                      JSR   $C8D4              ;EVALUATES EXP
580  0345  86D8                        STX   $D8                ;STORES X AT 216
590  0347  20F5BE                      JSR   $BEF5
600  034A  20D4C8                      JSR   $C8D4
610  034D  86C6                        STX   $C6                ;STORES X AT 198
620  034F  207FE0                      JSR   $E07F              ;POSITIONS CURSOR
630  0352  20F5BE                      JSR   $BEF5
640  0355  20D4C8                      JSR   $C8D4
645  0358  8A                          TXA
650  0359  F008                        BEQ   RETURN ;=0 THEN STR PRT
660  035B  A920                        LDA   #32                ;BLANKS
670  035D  20D2FF      BLANKA          JSR   $FFD2
680  0360  CA                          DEX
690  0361  D0FA                        BNE   BLANKA
700  0363  68          RETURN PLA
710  0364  A8                          TAY
720  0365  68                          PLA
730  0366  AA                          TAX
740  0367  68                          PLA
750  0368  60                          RTS
760  0369                              END
```

Hex dump of m/c code program

```
READY.

C*
     PC  IRQ  SR AC XR YR SP
.;  B780 E455 34 33 38 36 FA
.
.:   033A 48 8A 48 98 48 20 F5 BE
.:   0342 20 D4 C8 86 D8 20 F5 BE
.:   034A 20 D4 C8 86 C6 20 7F E0
.:   0352 20 F5 BE 20 D4 C8 8A F0
.:   035A 08 A9 20 20 D2 FF CA D0
.:   0362 FA 68 A8 68 AA 68 60 FF
.:   036A FF FF FF FF FF FF FF FF
.

.
READY.
```

M/C Code behind REM statements

The PET Basic Interpreter ignores REM statements and anything behind them. Thus numbers can be placed in the memory locations between the REM statement and the zero byte which denotes the end of line.

The best position to place the M/C code is at the beginning of the program as the program can be edited without the entry point of the M/C code being moved by the PET when the operator is editing the program.

For example if the programmer wants to add a line in the middle of a program the PET moves all lines following the added line up in memory to make room for the additional line. Thus all memory locations of lines above the added line will have been changed. Similarly if a line is edited the PET moves the basic program up or down in memory to compensate for additions or deletions. Also line number links are also changed.

If the M/C code segment is placed at the beginning of the program say at line $0 then subsequent editing or additions to the program will not change the M/C code segment — providing there are no zero bytes within the code — which the PET will interpret as end of basic line and play havoc with the complete program. If zero bytes are inevitable then the M/C code will have to be typed in when the program is completed and fully debugged.

The following explains how to locate the cursor positioning routine behind a REM statement at line $0.

```
   Type in the following short program line #0 with 47 '*' :-


0 REM ***********************************************
1 A$="TEST"
2 SYS1031,18,20,0:PRINTA$
3 FORI=1TO200:NEXT :REM DELAY LOOP
4 FORI=32768TO33767:POKEI,65
5 NEXT:REM FILL SCREEN WITH 'A'
6 FORI=1TO200:NEXT
7 SYS1031,18,20,255:REM BLANKS PORTION OF SCREEN


  Now type in


  sys4  <followed by a return>


  The following should appear - don't worry too much if the second line

  is not exactly as below -
```

```
 c*

    pc   irq   sr  ac  xr  yr  sp

  b780 e455 34  33  38  36  f8
```

The flashing cursor should be next to the dot. Now type in

```
 .m 0400 0460
```

The memory locations from $0400 (1024 decimal - start of basic) to $0460 should be displayed as follows:

```
.
.:  0400 00 37 04 00 00 8F 20 2A
.:  0408 2A 2A 2A 2A 2A 2A 2A 2A
.:  0410 2A 2A 2A 2A 2A 2A 2A 2A
.:  0418 2A 2A 2A 2A 2A 2A 2A 2A
.:  0420 2A 2A 2A 2A 2A 2A 2A 2A
.:  0428 2A 2A 2A 2A 2A 2A 2A 2A
.:  0430 2A 2A 2A 2A 2A 2A 00 45
.:  0438 04 01 00 41 24 B2 22 54
.:  0440 45 53 54 22 00 5B 04 02
.:  0448 00 9E 31 30 33 31 2C 31
.:  0450 38 2C 32 30 2C 30 3A 99
.:  0458 41 24 00 78 04 03 00 81
.:  0460 49 B2 31 A4 32 30 30 3A
```

The 8f is the hex representation of REM. The 20 following is the hex value of the space following the REM statement. The 2a's represent the stars (*) and it is in place of these we are going to type the M/C code.

Using the up cursor control key position the cursor over the first 2a and type in the hex numbers as shown below.

DON'T FORGET at the end of each block of hex numbers press 'return'.

```
.
.: 0400 00 37 04 00 00 8F 20 48
.: 0408 8A 48 98 48 20 F5 BE 20
.: 0410 D4 C8 86 D8 20 F5 BE 20
.: 0418 D4 C8 86 C6 20 7F E0 20
.: 0420 F5 BE 20 D4 C8 8A F0 08
.: 0428 A9 20 20 D2 FF CA D0 FA
.: 0430 68 A8 68 AA 68 60 00 45
.: 0438 04 01 00 41 24 B2 22 54
.: 0440 45 53 54 22 00 5B 04 02
.: 0448 00 9E 31 30 33 31 2C 31
.: 0450 38 2C 32 30 2C 30 3A 99
.: 0458 41 24 00 78 04 03 00 81
.: 0460 49 B2 31 A4 32 30 30 3A
.
.
```

When all the hex numbers have been typed in check them carefully and when satisfied type 'x' to bring you back to basic.

The listing of this program should look like the following :

```
 0 REM HRUNHPRINT#H PCOS APPENDLEFT$DIMRENAME PCOS APPENDLEFT$DIMASC
?OVERFLOW  COS APPENDLEFT$RUN(STEP   BACKUPπMID$HEADERF=(NOT(+(
 1 A$="TEST"
 2 SYS1031,18,20,0:PRINTA$
 3 FORI=1TO200:NEXT:REM DELAY LOOP
 4 FORI=32768TO33767:POKEI,65
 5 NEXT:REM FILL SCREEN WITH 'A'
 6 FORI=1TO200:NEXT
 7 SYS1031,18,20,255 :REM BLANKS PORTION OF SCREEN
READY.
```

Now save the program (just in case you have made a mistake in typing in the hex numbers) and then RUN the program.

I hope the ideas in the article will promote some interest in experimenting with the various routines and will prove of some use.

MASTER ...a new concept

MASTER is a totally new concept - a complete package for program and system development. MASTER adds 85 new commands to Commodore Basic IV, so that you can speedily and efficiently develop your own reliable software, whatever the application. With MASTER your programs can be up and running in double-quick time, and will out-perform standard Basic IV programs every time. One MASTER command can replace whole paragraphs of Basic code. With MASTER, projects that would have been non-starters can be tackled with ease. Add MASTER to your Pet - and let your Pet amaze you!

ISAM DISK FILES

MASTER has 17 commands to give keyed-access to your disk data. Data can be retrieved/added/deleted/updated from your files by ASCII key (up to 30 bytes long). You can read forwards or backwards through your files by key, starting anywhere. Or for extra fast access, you can read your files in creation order, even faster than reading a standard sequential file! For on-line data entry, file indexes can be updated optionally in batch mode. Up to 10 MASTER files can be opened at once, with no space restrictions except disk capacity. You can mix MASTER files with standard DOS files.

SCREEN GENERATOR

MASTER has 20 commands to give you complete control of the screen and keyboard input. Input/output can be done through screen zones, which may be formatted, eg for numerics/alpha/dates. Screen layouts (with zones) can be saved and loaded from disk, or can be swopped around within memory with the paging system. Windows can be declared, with full scrolling capabilities.

REPORT GENERATOR

MASTER has 10 commands to give you complete control of your printer output through report formats. Each report format can contain up to 128 output zones, which can be preformatted as required. Report formats can be saved and loaded from disk.

BASIC ENHANCEMENTS

MASTER has 18 commands of Basic enhancement including automatic date control - 20 place decimal precision arithmetic - data packing/unpacking to save space on disk - direct block access to disk - variable transfer to/from buffers - searching within strings - string conversion upper to lower, lower to upper case - GOTO and GOSUB with variables as line numbers - program scroll up and down - and a NOLIST feature to protect your programs from unauthorised listing.

BASIC ENHANCEMENTS (96K only)

MASTER has 17 extra commands in 96K mode (CBM 8096 or CBM 8032 upgraded or with CP/MAKER) including the "Toolkit" type commands AUTO, DELETE, RENUmber (all or part of program), DUMP, ERROR, FIND, TRACE/OFF (shows whole program line at top of screen). All of these except AUTO and ERROR can be included in programs. In addition there is CALL for calling your machine-code routines (while passing up to 15 parameters), FETCH for loading machine-code from disk (without disturbing Basic pointers), PLOT and RESET for medium resolution screen plotting (scale 50 by 150), PRINT USING for formatted output to screen or printer, IF ... THEN... ELSE ... for compact conditional programming, STOP KEY enable/disable, and HARDCOPY to dump the screen to printer (margin can be defined).

MEMORY MANAGEMENT (96K only)

MASTER gives total control from Basic over the 64K expansion memory. 26K is reserved for variables, which are maintained even while you edit or load programs. Up to 15 programs can be held simultaneously in the 46K program area, and you can switch from one to another automatically, or use GOTO or GOSUB and return to your main program.

HARDWARE SUPPORTED

MASTER is written in 6502 machine code and supports the CBM 8032, CBM 8032 with CBM 64K Expansion Memory or CP/MAKER, and CBM 8096, with CBM 4040 and CBM 8050 disks. MASTER is supplied with a comprehensive User Reference Manual, Quick Reference Guide, Diskette with demonstration programs, and a "dongle" which can be attached to either cassette port. Our support service is FREE, so we try to ensure that our documentation is as good as possible - so you don't need to bother us!

PM 96

PM96 for the CBM 8096 contains the Memory Management and 96K Basic Enhancements of MASTER, as a separate package, which includes a User Reference Manual, Quick Reference Guide, Diskette with demo programs, and Support Rom for the front (UD11) Expansion Rom slot.

Price List

MASTER Development System	£300.00
PM96 Basic Enhancement/Memory Management	£99.50
KRAM Keyed Disk Access (state which disk)	£86.95
COMMAND-O (Basic IV) (state which Pet)	£59.95
DISK-O-PRO (Basic II)	£59.95
SPACEMAKER 4-Rom adaptor (not 8096)	£29.95
PRONTO-PET hard/soft reset (Basic II)	£9.99

SPECIAL OFFERS

VISICALC 96K	RRP £180.00, less £35.00	£145.00
VISICALC 32K	RRP £159.00, less £30.00	£129.00
WORDPRO IV/V+	RRP £395.00, less £98.75	£296.25
WORDPRO III+	RRP £275.00, less £68.75	£206.25
WORDPRO II+	RRP £140.00, less £35.00	£105.00

ORDERING INFORMATION

Orders can be made by post, telephone or Prestel. ADD 15% VAT TO QUOTED PRICES. For same-day service, telephone 01-546-7256; we accept cheque/PO and ACCESS or BARCLAYCARD. (For over-the-counter sales please see your nearest Commodore Dealer). For Mail-order, write to our address below, or to our FREEPOST address (no stamp required) which is: Calco Software, FREEPOST, Kingston-upon-Thames, Surrey KT2 7BR. [Ref AD5

Calco Software

Lakeside House Kingston Hill
Surrey KT2 7QT (01) 546-7256

Basic Programs

The four function keys on the right hand side of the VIC are probably the most neglected part of the whole computer. Relegated to 'PRESS F1 TO START THE GAME', and dismissed with less than a page of text and a simple basic program by both the VIC REVEALED and the PROGRAMMERS REFERENCE GUIDE, the only way to use them as true definable function keys has been to spend £35 on one of Commodore's utility cartridges.

That is until now!

Using a simple 160 byte routine which sits at the top of BASIC memory, you can assign a separate function up to 8 characters long to each of the 8 keys.

This routine, which is loaded by a BASIC program, becomes part of the IRQ (Interrupt ReQuest) vector. This interrupt is the one which the processor calls 60 times a second to update the jiffy clock, scan the keyboard and check the RUN/STOP key.

Enter the program below and SAVE it before you RUN it! When the program has finished, it NEWs itself.

Now type RUN and press RETURN. All being well, the screen should clear and the messages ''FUNCTION KEYS DEFINED'' and ''READY.'' should appear, along with the cursor. If you get the message ''DATA ERROR'', then you have made a mistake entering the data statements in lines 10-120. CHECK THEM CAREFULLY!

Assuming that you have entered the program correctly, pressing the function keys should give you the following functions:

F1	LIST + CHRS(13)
F2	GOSUB
F3	RUN + CHRS (13)
F4	PRINT
F5	GOTO
F6	CHRS)
F7	LOAD
F8	RETURN + CHRS(13)

If you wish to change any of these functions, simply alter the DATA statements in lines 300-310. To eliminate the need to press carriage return, you can add one by simply entering' ' at the appropriate point(s).

For eaxmple: if you wish to LIST the program whenever a key is pressed, change the appropriate DATA statement to ''LIST ''.

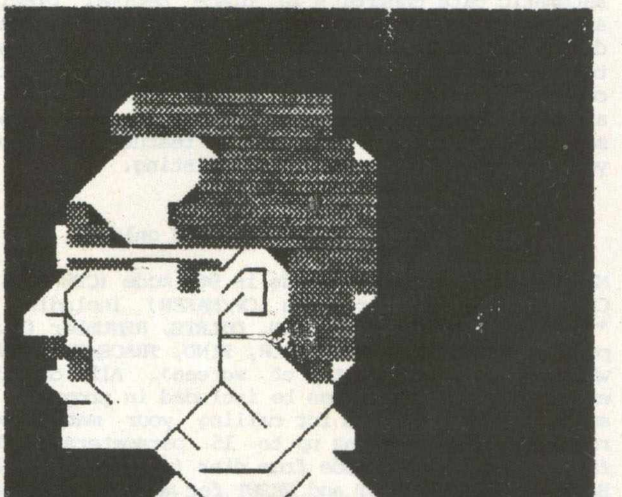
The function keys can be cleared by pressing RUN/STOP and RESTORE together. To re-enable the keys, enter 'SYS 7520'.

1 — You can only have up to 8 characters maximum per key. To enter longer commands use the abbreviations (such as P SHIFT 0 for POKE) listed in the manual, or allot parts to individual keys.

2 — To change a function, the program must be reloaded.

Finally, to disable the RUN/STOP key while the function keys are in operation, change the last three numbers in line 120 to 194, 234, 170

```
1 REM****************
2 REM*              *
3 REM* DEFINE  KEYS *
4 REM*              *
5 REM* BY DAVE TONG *
6 REM*              *
7 REM*  (C) 7/7/82  *
8 REM*              *
9 REM****************
10 DATA 120,169,128,141,20,3,169,29
20 DATA 141,21,3,88,133,56,169,96
30 DATA 133,55,96,160,64,169,0,153
40 DATA 191,29,136,208,250,96,234,234
50 DATA 72,138,72,152,72,165,197,197
60 DATA 251,240,44,133,251,41,39,201
70 DATA 39,208,36,24,165,251,42,41
80 DATA 240,172,141,2,240,3,24,105
90 DATA 8,105,128,133,252,169,29,133
100 DATA 253,160,0,177,252,153,119 2
110 DATA 200,192,8,208,246,132,198,104
120 DATA 168,104,170,104,76,191,234,170
199 REM      LOAD MACHINE CODE ROUTINES
200 POKE 55,96:POKE56,29:CLR:Z=0:FOR X=0 TO 95
210 READ Y:Z=Z+Y:POKE 7520+X,Y:NEXT X
220 IF Z<> 12270 THEN PRINT"DATA ERROR! RE-ENTER":STOP
230 SYS (7520):SYS (7539)
235 REM       SYS 7520 ACTIVATES THE KEYS
236 REM       SYS 7539 ERASES THE FUNCTIONS
240 FOR X=1 TO 8:READ N$
250 L=LEN(N$):IF L>8 THEN PRINTX;N$ PRINT"8 CHARACTERS MAXIMUM!":STOP
260 FOR Y=1 TO L:P=ASC(MID$(N$,Y,1)):IF P=95 THEN P=13
270 POKE 7607+Y+8*X,P:NEXT Y:NEXT X
280 PRINT"JFUNCTION KEYS DEFINED.":CLR:NEW
298 REM       PUT YOUR OWN FUNCTIONS HERE < MAXIMUM 8 CHARACTERS! >
299 REM       FOR CARRIAGE RETURN ENTER '←'
300 DATA "LIST←","GOSUB","RUN←","PRINT"
310 DATA "GOTO","CHR$(","LOAD","RETURN←"

READY.
```



DO VIC ROBOTS DREAM OF ELECTRIC SHEEP ?

```
 100 REM PET BENELUX
 110 REM EXCHANGE
 120 REM NETHERLANDS
 130 POKE36879,8:POKE36869,240
 140 PRINT"▀▜  *** VIC ROBOTS ***":PRINT"▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓▓
PRESS ANY KEY TO  STOP"
 150 GOTO190
 160 PRINT"▓▓▓▓";:FORI=1TOP%(P,0):PRINT"▓";:NEXT:FORI=1TOP%(P,1):PRINT
"▓";:NEXT
 170 PRINTD$(FNF(X))L$(FNF(X))R$(FNF(X))U$(FNF(X)):RETURN
 180 FORI=1TO10:GETH$:NEXT
 190 X=5:DIMD$(5),L$(5),R$(5),U$(5),P%(7,1):DEFFNF(X)=RND(1)*5+1:READX$
 200 FORI=1TOX:READD$(I):D$(I)=D$(I)+X$:NEXT:READX$:FORI=1TOX:READL$(I)
 210 L$(I)=L$(I)+X$:NEXT:READX$:FORI=1TOX:READR$(I):R$(I)=R$(I)+X$:NEXT
:READX$
 220 FORI=1TOX:READU$(I):U$(I)=U$(I)+X$:NEXT
 230 P%(0,1)=7:P%(0,0)=1:P%(1,1)=7:P%(1,0)=6
 240 P%(2,1)=7:P%(2,0)=11:P%(3,1)=7:P%(3,0)=16
 250 P%(4,1)=16:P%(4,0)=1:P%(5,1)=16:P%(5,0)=6
 260 P%(6,1)=16:P%(6,0)=11:P%(7,1)=16:P%(7,0)=16
 270 DATA" ▀█████","▘ ▞▙ ▀█████.▞▌▆ "," ▟▙ ▀█████ ▞▌▆ "
—▆"
 "

||█  "
 310 DATA" "," ▞▘▙ ▀█████"▞.,▟ ▀█████ ▞▘▙ ▀█████      "
 320 DATA" ▞ ▙ ▀█████"▞▚▆ ▀█████ ▞"▙ ▀█████ .. "
 330 DATA" ▞ ▙ ▀█████"▞▀▚▞▟ ▀█████ ▞▘▙ ▀█████ .. "
 340 DATA" ▞ ▙ ▀█████ ▞ ▚▟ ▀█████ ▄▞▆ ▀█████ .."
 350 DATA" ▞ ▙ ▀█████.▞ ▘▆ ▀█████ ▞▚▙ ▀█████    "
 360 K$="▆▜▙▚▓▆▜"
 370 P=INT(RND(1)*8):PRINTMID$(K$,P+1,1);:GOSUB160
 380 GETA$:IFA$<>""THEN400
 390 GOTO370
 400 PRINT"▀▆";:POKE36879,27:END
```

CHARACTER BUILDING EXERCISES

```
 100 REM PET BENELUX
 110 REM EXCHANGE
 120 REM NETHERLANDS
 130 POKE36879,42:PRINT"▀▜* CHARACTERGENERATOR *":POKE900,0
 140 POKE56,PEEK(56)-2:RUN150
 150 CS=256*PEEK(52)+PEEK(51)
 160 FORI=CSTOCS+511:POKEI,PEEK(I+32768-CS):NEXT
 170 PRINT"▀1RUN180":PRINT"RUN":POKE198,3:POKE631,19:POKE632,13:POKE633,
13:END
 180 S=7680:CL=22
 190 CS=256*PEEK(52)+PEEK(51)
 200 CR=0:LN=1000+PEEK(900):P=12:BG=1:BR=1
 210 POKE36879,42
 220 DEFFNA(XX)=S+R*CL+C:REM SCREEN POKE LOCATION
 230 DEFFNB(XX)=8*R+C:REM SCREEN POKE VALUE FOR CHARACTER
 240 GOTO600
```

Basic Programs

```
250 PRINT"]▓":GOSUB830
260 PRINT"▓";:FORI=0TO7:PRINT"........":NEXT:F=0
270 PRINT"▓":R=0:C=0
280 Z=FNA(0)
290 IFF=0THEN330
300 IFZ=ZLTHEN320
310 POKEZL,IL:ZL=Z:IL=PEEK(ZL)
320 POKEZ+30720,0
330 POKEZ+30720,0:GETA$:IFA$=""THEN330
340 POKEZ+30720,1
350 REM CURSOR CONTROL OPTIONS
360 IFA$="Q"THENPOKE56,PEEK(56)+2:POKE36869,240:PRINT"]":END
370 IFA$="▓"ANDC=7THENC=0:GOTO280
380 IFA$="▓"THENC=C+1:GOTO280
390 IFA$="▓"ANDC=0THENC=7:GOTO280
400 IFA$="▓"THENC=C-1:GOTO280
410 IFA$="▓"ANDR=7THENR=0:GOTO280
420 IFA$="▓"THENR=R+1:GOTO280
430 IFA$="]"ANDR=0THENR=7:GOTO280
440 IFA$="]"THENR=R-1:GOTO280
450 IFA$="▓"THEN270
460 IFF=1THEN560
470 REM DEFINE NEW CHARACTER OPTIONS
480 IFA$="+"THENPOKEZ,81:GOTO280
490 IFA$="-"THENPOKEZ,46:GOTO280
500 IFA$="="THEN700
510 IFA$="]"THEN260
520 IFA$="R"THEN600
530 IFA$="B"THEN790
540 GOTO280
550 REM REVIEW CHARACTER SET OPTIONS
560 CR=FNB(0)
570 IFA$="N"THENPOKE36869,240:GOTO250
580 IFA$="E"THENPOKE36869,240:F=0:GOTO750
590 GOTO280
600 POKE36869,255:R=4:C=0:ZL=FNA(0):IL=32
610 PRINT"]▓▓ABCDEFG":PRINT"HIJKLMNO":PRINT"PQRSTUVW":PRINT"XYZ[\]↑←"
:F=1
620 PRINT" !"+CHR$(34)+"#$%&'":PRINT"()*+,-./":PRINT"01234567":PRINT"
89:;<=>?"
630 PRINT"▓"SPC(12);"▓OPRTIONS▓":PRINT
640 PRINTSPC(10);"▓N NEW CHAR▓"
650 PRINTSPC(10);"▓E EDIT CHAR▓"
660 PRINTSPC(10);"▓Q QUIT▓"
670 BC=PEEK(38400)
680 GOTO280
690 REM UPDATE CHARACTER DATA IN TABLE
700 PRINT"▓";:X=CS+8*CR:FORR=0TO7:SM=0:FORC=0TO7:D=7-C
710 SM=SM-2↑D*(PEEK(FNA(0))=81):NEXTC
720 POKEX+R,SM:PRINTSPC(8);SM:NEXTR
730 R=0:C=0:GOTO280
740 REM EDIT CHARACTER FROM TABLE
750 X=CS+8*CR:PRINT"]":FORR=0TO7:Y=PEEK(X+R):FORC=0TO7:Z=FNA(0)
760 Q=46:Y=Y*2:IFY>255THENQ=81:Y=Y-256
770 POKEZ,Q:NEXTC,R:R=0:C=0:GOSUB830:GOTO280
780 REM BASIC STATEMENTS TO DEFINE CHARACTER
790 X=CS+8*CR:PRINT"▓▓▓▓▓▓▓▓▓▓"
800 PRINTLN;"DATA";RIGHT$(STR$(X),-1+LEN(STR$(X)));:FORI=XTOX+7
```

How to make the best home computer in the world even better.

Peripherals to turn a powerful computer into a super-computer for the professional.

With VIC, you have the finest home computer money can buy. And the more you use it, the more you will ask it to do.

Pretty soon, you'll want to extend VIC's vast potential to the full; and there is a wide range of VIC peripherals to help you do it.

Disk drives, disk-based software, a printer, cassette unit, joysticks, paddles–with these, VIC computing becomes total computing: giving you true professional power and capability.

We describe the major units here.

VIC PRINTER

The VIC Printer, like all VIC peripherals, offers a very high specification at a very competitive price.

It will print programs, letters, business data, graphic displays and so on.

Its main features include: 80 characters per line • Tractor feed dot matrix • 30 characters per second print speed • Full alphanumerics and graphic printing • Double-size character capability • All cables and leads.

VIC FLOPPY DISK UNIT

The VIC single-drive Disk Unit provides a fast, accurate and efficient means of storing and retrieving data and programs.

Together with the Printer, it transforms the VIC 20 into the ideal system for the small businessman or serious computer programmer.

Features include: 174,848 bytes capacity • Uses soft-sectored standard 5¼″ single density floppy disks • Direct interface to VIC • Direct compatibility with Printer Intelligent system independent of VIC. (VIC RAM not required to run it).

EXPANSION MEMORY CARTRIDGES

Special plug-in cartridges are available to expand VIC's memory. 3K, 8K and 16K RAM packs plug directly into the computer.

A Memory Expansion Board is also available to develop VIC's capabilities to the maximum.

For full details of VIC 20, its peripherals and software, and a list of your local dealers, contact:

The Commodore Information Centre,
675 Ajax Avenue, Slough, Berkshire,
SL1 4BG. Tel: Slough 79292.

commodore

VIC 20

The best home computer in the world.

Basic Programs

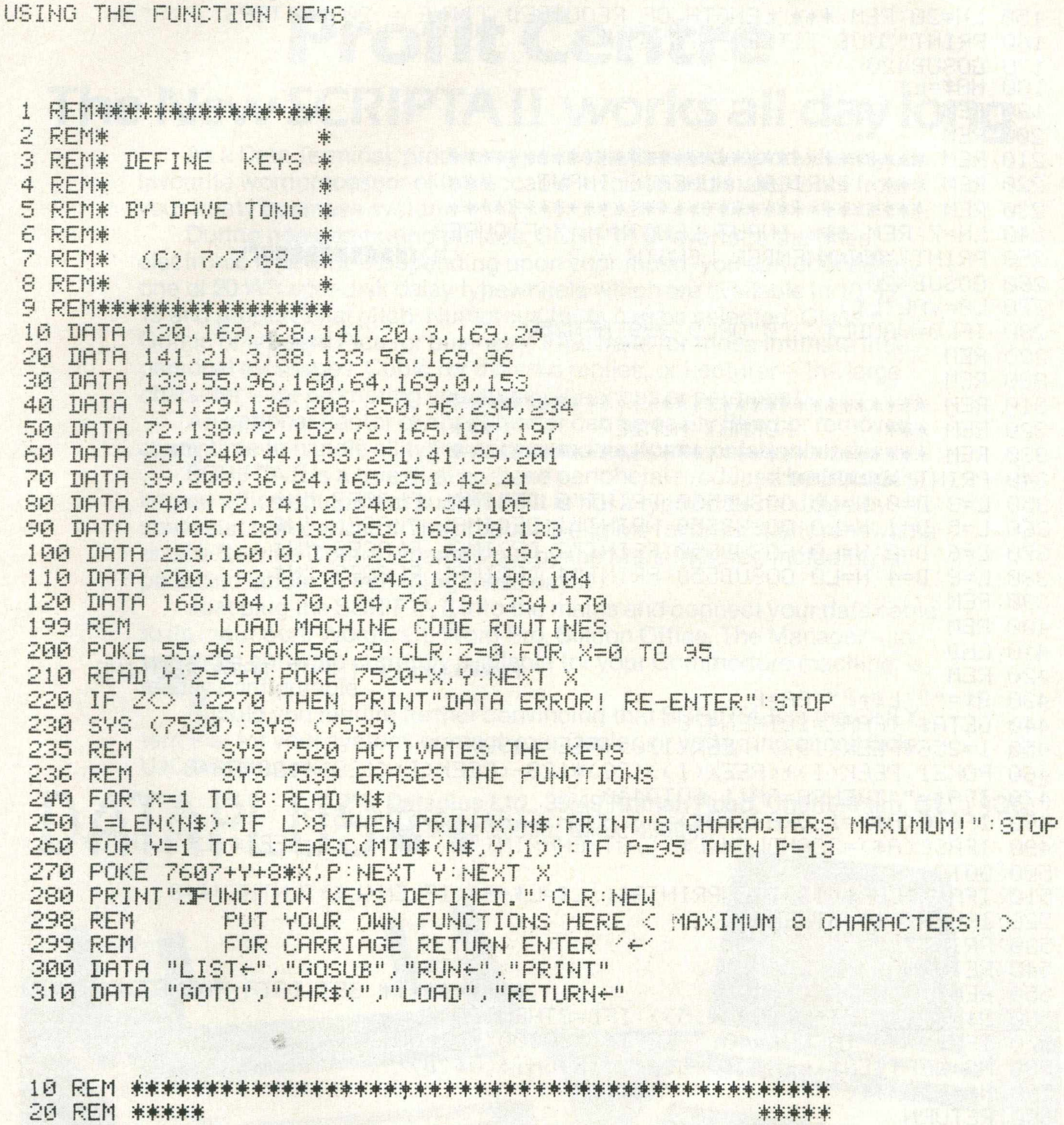
```
810 PRINT",";RIGHT$(STR$(PEEK(I)),-1+LEN(STR$(PEEK(I))));:NEXTI:PRINT:
PRINT"RUN ●"
820 POKE900,PEEK(900)+1:POKE198,9:FORI=0TO8:POKEI+631,13:NEXT:END
830 PRINT"■■";SPC(13)"■OPTIONS■":PRINT
840 PRINTSPC(P);"■+■ ADD DOT"
850 PRINTSPC(P);"■-■ ERASE"
860 PRINTSPC(P);"■=■ UPDATE"
870 PRINTSPC(P);"■R■ REVIEW"
880 PRINTSPC(P);"■Q■ QUIT"
890 RETURN
```

CIRCLE DEMO - VIC 20

```
 10 POKE36879,42:PRINT"■■*** VIC CIRCLEDEMO ***
 15 POKE36869,253:POKE36867,PEEK(36867)OR128
 20 POKE55,0:POKE56,19:POKE51,0:POKE52,19:CLR:S=32768:T=5120
 25 FORI=0TO255*8+7:POKEI+T,PEEK(I+S):NEXT:GOTO55
 30 X%=X/8:Y%=Y/8:P=X%+Y%*22+7680
 35 Q=PEEK(P):IFQ<128THEN45
 40 C=5120+Q*8+(YAND7):POKEC,PEEK(C)OR(2↑(7-(XAND7))):RETURN
 45 CN=CN+1:S=5120+(127+CN)*8:T=5120+Q*8
 50 FORI=0TO7:POKES+I,PEEK(T+I):NEXT:Q=127+CN:POKEP,Q:GOTO40
 55 FORI=1TO22:POKE7680+22*I,93:NEXT
 60 POKE7680+11*22,107
 65 FORI=1TO21:POKE7680+11*22+I,64:NEXT
 70 RD=40:FORZ=0TO2*πSTEP.05:X=COS(Z)*RD+80:Y=SIN(Z)*RD*1.7+88:GOSUB30
 75 X=COS(Z)*RD+90:Y=SIN(Z)*RD*1.7+98:GOSUB30:NEXT
 80 GETA$:IFA$=""THEN80
 85 POKE36879,27:PRINT"■■";:POKE36869,240
```

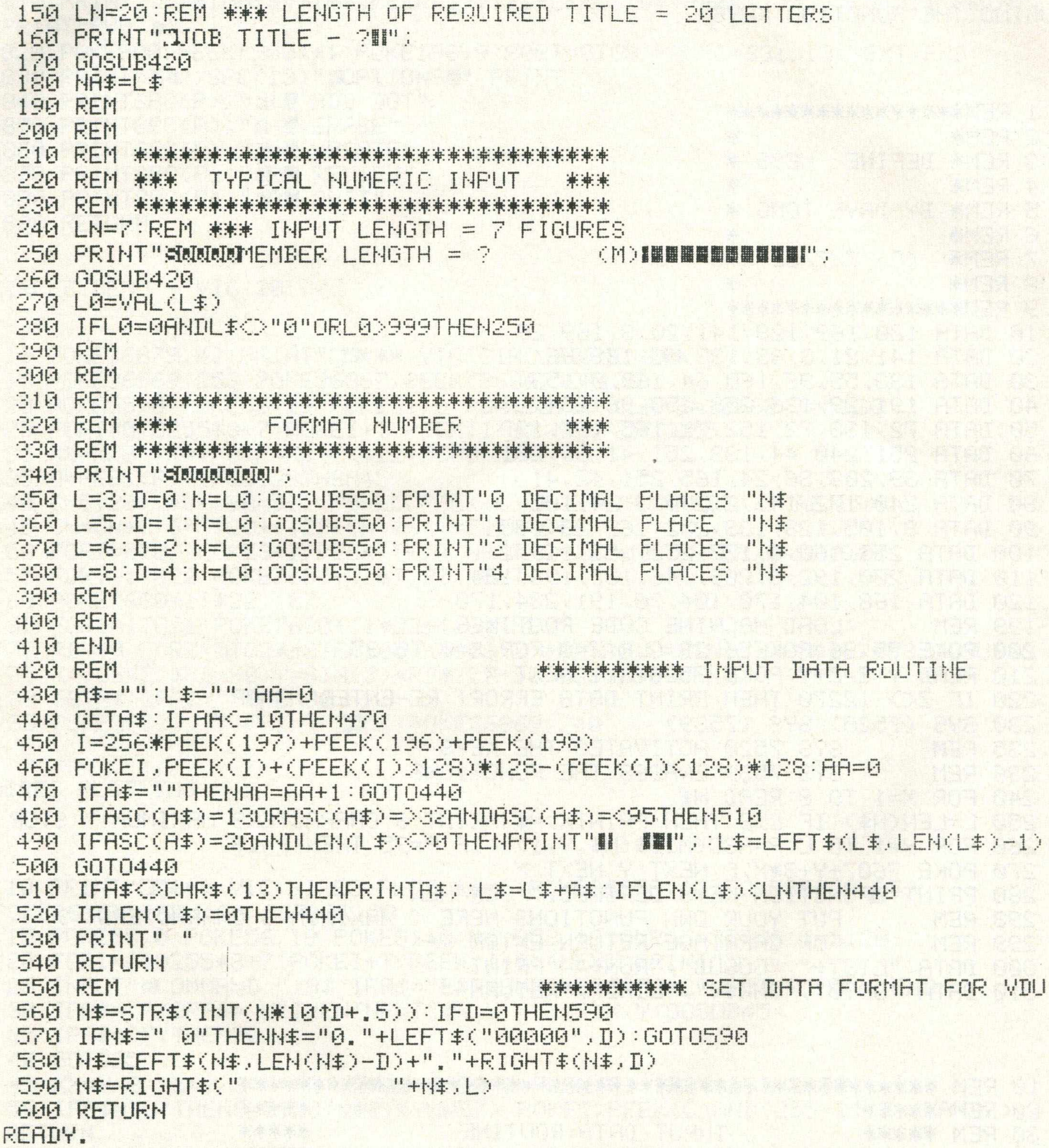
HIRES PLOTTING

```
 10 PRINT"■■*** VIC HIRES-PLOT ***":POKE36879,42:POKE36869,253:
POKE36867,PEEK(36 867)OR128
 15 POKE55,0:POKE56,19:POKE51,0:POKE52,19:CLR:S=32768:T=5120
 20 FORI=0TO255*8+7:POKEI+T,PEEK(I+S):NEXT:PRINT"■
 25 INPUT"■COMMAND ";A$:IFA$="E"THENPOKE36879,27:PRINT"■■";:END
 30 IFA$="P"ORA$="U"THENINPUT"■X,Y          ";X,Y:GOSUB45
 35 IFA$="F"THEN70
 40 GOTO25
 45 X%=X/8:Y%=Y/8:P=X%+Y%*22+7680:Q=PEEK(P):IFQ<128THEN60
 50 IFA$="U"THENC=5120+Q*8+(YAND7):POKEC,PEEK(C)AND(255-2↑(7-(XAND7)))
:RETURN
 55 C=5120+Q*8+(YAND7):POKEC,PEEK(C)OR(2↑(7-(XAND7))):RETURN
 60 CN=CN+1:S=5120+(127+CN)*8:T=5120+Q*8:FORI=0TO7:POKES+I,PEEK(T+I)
:NEXT
 65 Q=127+CN:POKEP,Q:GOTO55
 70 PRINT"■":FORI=0TO22:POKE7680+22*I,93:NEXT:POKE7680+11*22,107
:DATA15,23,19
  75 FORI=1TO21:POKE7680+11*22+I,64:NEXT
  80 FORZ=1TO3:READB:FORX=4TO175:Y=.5*X+50+SIN((X-4)/176*B*π)*15:GOSUB45
:NEXT:NEXT:GOTO25
```



```
USING THE FUNCTION KEYS


1 REM****************
2 REM*              *
3 REM* DEFINE  KEYS *
4 REM*              *
5 REM* BY DAVE TONG *
6 REM*              *
7 REM*  (C) 7/7/82  *
8 REM*              *
9 REM****************
10 DATA 120,169,128,141,20,3,169,29
20 DATA 141,21,3,88,133,56,169,96
30 DATA 133,55,96,160,64,169,0,153
40 DATA 191,29,136,208,250,96,234,234
50 DATA 72,138,72,152,72,165,197,197
60 DATA 251,240,44,133,251,41,39,201
70 DATA 39,208,36,24,165,251,42,41
80 DATA 240,172,141,2,240,3,24,105
90 DATA 8,105,128,133,252,169,29,133
100 DATA 253,160,0,177,252,153,119,2
110 DATA 200,192,8,208,246,132,198,104
120 DATA 168,104,170,104,76,191,234,170
199 REM      LOAD MACHINE CODE ROUTINES
200 POKE 55,96:POKE56,29:CLR:Z=0:FOR X=0 TO 95
210 READ Y:Z=Z+Y:POKE 7520+X,Y:NEXT X
220 IF Z<> 12270 THEN PRINT"DATA ERROR! RE-ENTER":STOP
230 SYS (7520):SYS (7539)
235 REM      SYS 7520 ACTIVATES THE KEYS
236 REM      SYS 7539 ERASES THE FUNCTIONS
240 FOR X=1 TO 8:READ N$
250 L=LEN(N$):IF L>8 THEN PRINTX;N$:PRINT"8 CHARACTERS MAXIMUM!":STOP
260 FOR Y=1 TO L:P=ASC(MID$(N$,Y,1)):IF P=95 THEN P=13
270 POKE 7607+Y+8*X,P:NEXT Y:NEXT X
280 PRINT"⊐FUNCTION KEYS DEFINED.":CLR:NEW
298 REM      PUT YOUR OWN FUNCTIONS HERE < MAXIMUM 8 CHARACTERS! >
299 REM      FOR CARRIAGE RETURN ENTER '←'
300 DATA "LIST←","GOSUB","RUN←","PRINT"
310 DATA "GOTO","CHR$(","LOAD","RETURN←"
```

```
 10 REM *********************************************
 20 REM *****                                   *****
 30 REM *****        INPUT DATA ROUTINE         *****
 40 REM *****               AND                 *****
 50 REM *****        DATA FORMAT ROUTINE        *****
 60 REM *****               BY                  *****
 70 REM *****            J.A. EVANS             *****
 80 REM *****                                   *****
 90 REM *********************************************
100 REM
110 REM
120 REM ********************************
130 REM ***    TYPICAL STRING INPUT  ***
140 REM ********************************
```



```
 150 LN=20:REM *** LENGTH OF REQUIRED TITLE = 20 LETTERS
 160 PRINT"]JOB TITLE - ?▮▮";
 170 GOSUB420
 180 NA$=L$
 190 REM
 200 REM
 210 REM ********************************
 220 REM ***  TYPICAL NUMERIC INPUT   ***
 230 REM ********************************
 240 LN=7:REM *** INPUT LENGTH = 7 FIGURES
 250 PRINT"SQQQQMEMBER LENGTH = ?         (M)▮▮▮▮▮▮▮▮▮▮▮▮";
 260 GOSUB420
 270 L0=VAL(L$)
 280 IFL0=0ANDL$<>"0"ORL0>999THEN250
 290 REM
 300 REM
 310 REM ********************************
 320 REM ***        FORMAT NUMBER     ***
 330 REM ********************************
 340 PRINT"SQQQQQQ";
 350 L=3:D=0:N=L0:GOSUB550:PRINT"0 DECIMAL PLACES "N$
 360 L=5:D=1:N=L0:GOSUB550:PRINT"1 DECIMAL PLACE  "N$
 370 L=6:D=2:N=L0:GOSUB550:PRINT"2 DECIMAL PLACES "N$
 380 L=8:D=4:N=L0:GOSUB550:PRINT"4 DECIMAL PLACES "N$
 390 REM
 400 REM
 410 END
 420 REM                         ********** INPUT DATA ROUTINE
 430 A$="":L$="":AA=0
 440 GETA$:IFAA<=10THEN470
 450 I=256*PEEK(197)+PEEK(196)+PEEK(198)
 460 POKEI,PEEK(I)+(PEEK(I)>128)*128-(PEEK(I)<128)*128:AA=0
 470 IFA$=""THENAA=AA+1:GOTO440
 480 IFASC(A$)=13ORASC(A$)=>32ANDASC(A$)=<95THEN510
 490 IFASC(A$)=20ANDLEN(L$)<>0THENPRINT"▮  ▮▮";:L$=LEFT$(L$,LEN(L$)-1)
 500 GOTO440
 510 IFA$<>CHR$(13)THENPRINTA$;:L$=L$+A$:IFLEN(L$)<LNTHEN440
 520 IFLEN(L$)=0THEN440
 530 PRINT" "
 540 RETURN
 550 REM                         ********** SET DATA FORMAT FOR VDU
 560 N$=STR$(INT(N*10↑D+.5)):IFD=0THEN590
 570 IFN$=" 0"THENN$="0."+LEFT$("00000",D):GOTO590
 580 N$=LEFT$(N$,LEN(N$)-D)+"."+RIGHT$(N$,D)
 590 N$=RIGHT$("          "+N$,L)
 600 RETURN
READY.
```

Turn your Commodore into a full time Profit Centre

The New SCRIPTA II works all day long~

As a Data Terminal, producing perfect letters and reports from your favourite wordprocessor or impeccable invoices and statements from your trusty business system.

During non-computing periods, SCRIPTA II reverts to the finest electronic typewriter. Depending upon your mood, you can choose any one of 20 Whisper-disk daisy typewheels which are available in 10, 12, 15 and proportional pitch. Numerous fonts can be selected. Classic Gothic or Modern Pica for business, Fine Italic for those intimate little personal messages, Cubic for the chic replies, or Lecturer – the large character – for prompting you during speeches or sermons!

An optional tractor or sheet feeder can be easily fitted or removed at any time to handle all types of documents, forms or letter heads.

SCRIPTA II is a true dual purpose peripheral produced by Europe's largest office equipment supplier – it is not a typewriter modified by some electronic wizard! The product is eligible for an annually renewable on-site Service Contract carried out by the Manufacturer, including all parts and labour. A rare thing these days!

Just plug the SCRIPTA II into the mains and connect your data cable to its input port. Wordcraft, Word-Pro, Silicon Office, The Manager – in fact most software currently available for your Commodore machine, is readily configurable.

Should you require further convincing that SCRIPTA II is the ONLY terminal for your system, contact your Dealer, or write, ring or call Sole U.K. Distributor –

DATAPLUS

Dataplus Ltd., 39/49 Roman Road, Cheltenham, GL51 8QQ,
Tel: 0242-30030, Telex: 43594.

M/C Programming

SUPERMON for the VIC

Introduction

No machine language monitor is provided in the VIC ROMs. Serious users find themselves in the same position as the first generation (BASIC 1.0) of PET users. Since only a commercial product, VICMON, is available, the need for a public-domain utility seemed a worthwhile project.

Jim Butterfield has already developed TINYMON1, which may be found in the June issue of Commodore Computing. This offers the equivalent to the Commodore TIM monitor.

We've already been accustomed to the benefits inherent in Supermon, Extramon and Micromon for the PET/CBM. Herewith my adaptation of Suprmon for the VIC.

While we are on the subject of Jim Butterfield, I once again offer my thanks for his many contributions to the PET/CBM/VIC community. His splendid work and the donation of same to the public domain is quite remarkable. We often fail to acknowledge how rich is our store of knowledge because of this gentleman.

Features

Like its predecessors, VIC-20 SUPERMON loads and self-relocates to the top of VIC memory, regardless of the memory configuration installed. It was designed to fit in less than 2K, and I surrendered the ''single-step'' mode to accomplish this goal.

Make no mistake, there's a lot of time to be invested to reproduce the code of this program.

I've copied Jim's technique of ''absolute-load'' files with VIC-20 SUPERMON. This means that if you SAVE memory with the monitor, the VIC won't relocate it on you when you come to reLOAD the code.

'Jim said that the PET would ignore such files, but my utility called VICLOAD covers that shortcoming.

Procedure

Apart from the investment of time, you don't really need to be familiar with machine-launguage to enter the program. One of Supermon's benefits is its value for beginners in ML programming.

You'll need a PET/CBM with Upgrade (BASIC 2.0) or BASIC 4.0 ROM installed. Start with a freshly powered-up machine. Enter the ML monitor by typing 'SYS4' and hitting 'RETURN'.

There are 20 screens of information to be entered before you're done. Each requires the same procedure. Only the first (line entry) is slightly different:

1. Immediately after the ''.'', type the range of memory to be displayed, like so:

```
.M 0028 0028   RETURN
```

One line of hexadecimal digits is shown on the screen.

2. Move the cursor back to this line and type over the digits with the values:

```
.: 0028 01 04 DF OD DF OD DF OD  RETURN
```

Don't forget to hit the 'RETURN' key at the end of each line, or else the new values won't be remembered.

Now type in the starting and ending addresses for each of the 20 screens to be changed. For example, the first would be:

```
.M 0400 0478   RETURN
```

Copy the values shown in the accompanying ''hex dump'' for each block shown. Type right over the existing values, (which will probably show as ''AA'' on your screen). Do your best to double-check before proceding with the next block.

Repeat until the block from $OD80 to $ODF8 is done. Now exit the monitor, with:

```
.X   RETURN
```

NOW SAVE AND VERIFY THE PROGRAM. Do not pass ''GO'', do not collect $200 ... do this first!

Checking

Because of the 2000+ entries you've made, the chance for error is high. Several ''immediate-'' or ''Calculator- Mode'' statements are provided to verify your work.

These statements do a ''check-sum'' on the total program, each multiple of 5-screens (4 check-sums), and each ''line'' of 8-entries.

Enter the statements shown, with no line numbers please! On hitting 'RETURN', the total will be shown immediately below.

If your first one shows ''283370'', then it's probably perfect. Go to the head of the class, and start using VIC-20 SUPERMON.

Otherwise, work your way through the next four, noting which are correct. Each of these totals are a composite of 80 lines of entry, (or 5-screens, as we put them in).

For any incorrect block from above, there is a corresponding statement to type. This will give the individual totals for each line of entry. Mark the lines that are different. You will now have to re-enter the ML monitor and make the necessary corrections. ReSAVE this version and re-do the checksum until it's correct.

Operation

Disconnect your cassette recorder (power off the PET first, please). Reconnect to the VIC, turn everything on, LOAD VIC-20 SUPERMON and RUN it.

You should be greeted with the so-called ''Register Display'', as on the PET/CBM. Unless it's a B & W display, you'll see several usages of VIC colour.

Any entries you make will be in blue, while the VIC ..variously displays red for register headings or errors, purple for normal addresses, black for memory bytes and green for disassembly mnemonics or ''next'' addresses when assembling code.

Since we are dealing with a 22-character wide screen, the disassembly consumes two lines. The mnemonics are pushed to the right on the second line. More locations would have fit, but I chose to keep the actual ''bytes'' display. This of course permits you to change them, and causes an automatic redisplay of the same range of addresses.

```
 .
 .:   0580 85 1D A9 90 20 D2 FF A0
 .:   0588 00 00 20 CB F8 00 B1 C1
 .:   0590 20 34 FA 00 20 D3 F8 00
 .:   0598 C6 1D D0 F1 60 20 8B FA
 .:   05A0 00 90 0B A2 00 00 81 C1
 .:   05A8 C1 C1 F0 03 4C FC FA 00
 .:   05B0 20 D3 F8 00 C6 1D 60 A9
 .:   05B8 02 85 C1 A9 00 00 85 C2
 .:   05C0 60 98 48 20 CE F8 00 A9
 .:   05C8 9C 20 D2 FF 68 A2 2E 4C
 .:   05D0 43 FA 00 A9 20 2C A9 0D
 .:   05D8 4C D2 FF E6 C1 D0 06 E6
 .:   05E0 C2 D0 02 E6 26 60 A9 1C
 .:   05E8 20 D2 FF A2 00 00 BD EA
 .:   05F0 FF 00 20 D2 FF E8 E0 16
 .:   05F8 D0 F5 A0 3B 20 BB F8 00
 .
 .
 .:   0600 A5 00 00 20 34 FA 00 A5
 .:   0608 01 20 34 FA 00 20 B2 F8
 .:   0610 00 20 81 F8 00 F0 57 20
 .:   0618 BB FA 00 20 7C FA 00 90
 .:   0620 2E 20 6C FA 00 20 BB FA
 .:   0628 00 20 7C FA 00 90 23 20
 .:   0630 6C FA 00 20 70 F7 F0 3C
 .:   0638 A6 26 D0 38 A5 C3 C5 C1
 .:   0640 A5 C4 E5 C2 90 2E A0 3A
 .:   0648 20 BB F8 00 20 2D FA 00
 .:   0650 20 81 F8 00 F0 E0 4C FC
 .:   0658 FA 00 20 7C FA 00 90 03
 .:   0660 20 78 F8 00 20 B2 F8 00
 .:   0668 F0 05 20 7C FA 00 90 EB
 .:   0670 A9 05 85 1D 20 BB FA 00
 .:   0678 20 9C F8 00 D0 F8 4C 44
```

The goal of ''under 2K'' was met (by 3 bytes), but the single-step had to be sacrificed. Someone else may re-work Micromon for VIC, but its 4K size is of no use to the ''3583 BYTES FREE'' crowd.

In Closing

'If I use this half as much as Supermon for PET, it will have been worth the adaptation effort. Without Supermon 1.0 (for Original ROM), this would not have been feasible. A deep bow towards 14 Brooklyn Avenue, Toronto.

David A. Hook
58 Steel Street
BARRIE,
Ontario
L4M 2E9
(705) 726-8126

```
 .
 .:   0400 00 1A 04 64 00 99 22 93
 .:   0408 12 1D 1D 1D 1D 53 55 50
 .:   0410 45 52 20 56 49 43 4D 4F
 .:   0418 4E 00 2F 04 6E 00 99 22
 .:   0420 11 44 41 56 49 44 20 41
 .:   0428 2E 20 48 4F 4F 4B 00 44
 .:   0430 04 70 00 99 22 11 46 52
 .:   0438 4F 4D 20 53 55 50 45 52
 .:   0440 4D 4F 4E 00 5E 04 73 00
 .:   0448 99 22 11 42 59 20 4A 49
 .:   0450 4D 20 42 55 54 54 45 52
 .:   0458 46 49 45 4C 44 00 79 04
 .:   0460 78 00 9E 28 C2 28 34 33
 .:   0468 29 AA 32 35 36 AC C2 28
 .:   0470 34 34 29 AA 31 32 37 29
 .:   0478 00 00 00 AA AA AA AA AA
 .
 .
 .:   0480 A5 2D 85 22 A5 2E 85 23
 .:   0488 A5 37 85 24 A5 38 85 25
 .:   0490 A0 00 A5 22 D0 02 C6 23
 .:   0498 C6 22 B1 22 D0 3C A5 22
 .:   04A0 D0 02 C6 23 C6 22 B1 22
 .:   04A8 F0 21 85 26 A5 22 D0 02
 .:   04B0 C6 23 C6 22 B1 22 18 65
 .:   04B8 24 AA A5 26 65 25 48 A5
 .:   04C0 37 D0 02 C6 38 C6 37 68
 .:   04C8 91 37 8A 48 A5 37 D0 02
 .:   04D0 C6 38 C6 37 68 91 37 18
 .:   04D8 90 B6 C9 BF D0 ED A5 37
 .:   04E0 85 33 A5 38 85 34 6C 37
 .:   04E8 00 AA AA AA AA AA AA AA
 .:   04F0 BF 78 AD E8 FF 00 8D 16
 .:   04F8 03 AD E9 FF 00 8D 17 03
```

M/C Programming

```
.:  0500 A9 80 20 90 FF 58 00 00
.:  0508 D8 68 85 05 68 85 04 68
.:  0510 85 03 68 85 02 68 AA 68
.:  0518 A8 38 8A E9 02 85 01 98
.:  0520 E9 00 00 85 00 00 BA 86
.:  0528 06 20 CE F8 00 A2 42 A9
.:  0530 2A 20 43 FA 00 A9 52 D0
.:  0538 1B A9 1F 20 D2 FF A9 00
.:  0540 00 85 26 A2 0D A9 2E 20
.:  0548 43 FA 00 20 BB FA 00 C9
.:  0550 2E F0 F9 C9 20 F0 F5 A2
.:  0558 0E DD BB FF 00 D0 0C 8A
.:  0560 0A AA BD CB FF 00 48 BD
.:  0568 CA FF 00 48 60 CA 10 EC
.:  0570 4C FC FA 00 A5 C1 85 01
.:  0578 A5 C2 85 00 00 60 A9 05
```

```
.:  0700 C9 03 F0 E6 85 BA 20 CF
.:  0708 FF C9 0D 60 4C FC FA 00
.:  0710 20 8F F9 00 D0 F8 20 4F
.:  0718 F5 20 4C FA 00 A5 90 29
.:  0720 10 D0 EC 4C 44 F8 00 20
.:  0728 8F F9 00 C9 2C D0 E2 20
.:  0730 7C FA 00 20 6C FA 00 20
.:  0738 CF FF C9 2C D0 D5 20 7C
.:  0740 FA 00 A5 C1 85 AE A5 C2
.:  0748 85 AF 20 6C FA 00 20 CF
.:  0750 FF C9 0D D0 C0 20 82 F6
.:  0758 4C 44 F8 00 A5 C2 20 34
.:  0760 FA 00 A5 C1 48 4A 4A 4A
.:  0768 4A 20 63 FA 00 AA 68 29
.:  0770 0F 20 63 FA 00 48 8A 20
.:  0778 D2 FF 68 4C D2 FF 20 55
```

```
.:  0780 FA 00 2C 2D 91 30 F8 60
.:  0788 20 70 F7 D0 08 A9 03 85
.:  0790 9A A9 00 00 85 99 60 09
.:  0798 30 C9 3A 90 02 69 06 60
.:  07A0 A2 02 B5 C0 48 B5 C2 95
.:  07A8 C0 68 95 C2 CA D0 F3 60
.:  07B0 20 8B FA 00 90 02 85 C2
.:  07B8 20 8B FA 00 90 02 85 C1
.:  07C0 60 A9 00 00 85 2A 20 BB
.:  07C8 FA 00 C9 20 D0 09 20 BB
.:  07D0 FA 00 C9 20 D0 0E 18 60
.:  07D8 20 B2 FA 00 0A 0A 0A 0A
.:  07E0 85 2A 20 BB FA 00 20 B2
.:  07E8 FA 00 05 2A 38 60 C9 3A
.:  07F0 90 02 69 08 29 0F 60 20
.:  07F8 CF FF C9 0D D0 F8 68 68
```

```
.:  0680 F8 00 20 CF FF C9 0D F0
.:  0688 0C C9 20 D0 D1 20 7C FA
.:  0690 00 90 03 20 78 F8 00 A6
.:  0698 06 9A 78 A5 00 00 48 A5
.:  06A0 01 48 A5 02 48 A5 03 A6
.:  06A8 04 A4 05 40 78 A6 06 9A
.:  06B0 6C 02 C0 A0 01 84 BA 84
.:  06B8 B9 88 84 B7 84 90 84 93
.:  06C0 A9 40 85 BB A9 02 85 BC
.:  06C8 20 CF FF C9 20 F0 F9 C9
.:  06D0 0D F0 37 C9 22 D0 34 20
.:  06D8 CF FF C9 22 F0 0F C9 0D
.:  06E0 F0 28 91 BB E6 B7 C8 C0
.:  06E8 10 F0 20 D0 EA 20 CF FF
.:  06F0 C9 0D F0 16 C9 2C D0 DD
.:  06F8 20 8B FA 00 29 0F F0 EA
```

```
.:  0880 C1 85 1E 98 E5 C2 A8 05
.:  0888 1E 60 20 E3 FA 00 20 6C
.:  0890 FA 00 20 F4 FA 00 20 1A
.:  0898 FB 00 20 F4 FA 00 20 31
.:  08A0 FB 00 20 6C FA 00 90 15
.:  08A8 A6 26 D0 64 20 2A FB 00
.:  08B0 90 5F A1 C1 81 C3 20 13
.:  08B8 FB 00 20 D3 F8 00 D0 EB
.:  08C0 20 2A FB 00 18 A5 1E 65
.:  08C8 C3 85 C3 98 65 C4 85 C4
.:  08D0 20 1A FB 00 A6 26 D0 3D
.:  08D8 A1 C1 81 C3 20 2A FB 00
.:  08E0 B0 34 20 C7 FA 00 20 CA
.:  08E8 FA 00 4C 7F FB 00 20 E3
.:  08F0 FA 00 20 6C FA 00 20 F4
.:  08F8 FA 00 20 6C FA 00 20 BB
```

```
.:  0900 FA 00 20 8B FA 00 90 14
.:  0908 85 1D A6 26 D0 11 20 31
.:  0910 FB 00 90 0C A5 1D 81 C1
.:  0918 20 D3 F8 00 D0 EE 4C FC
.:  0920 FA 00 4C 44 F8 00 20 E3
.:  0928 FA 00 20 6C FA 00 20 F4
.:  0930 FA 00 20 6C FA 00 20 BB
.:  0938 FA 00 A2 00 00 20 BB FA
.:  0940 00 C9 27 D0 14 20 BB FA
.:  0948 00 9D 10 02 E8 20 CF FF
.:  0950 C9 0D F0 22 E0 20 D0 F1
.:  0958 F0 1C 8E 00 00 01 20 92
.:  0960 FA 00 90 C6 9D 10 02 E8
.:  0968 20 CF FF C9 0D F0 09 20
.:  0970 8B FA 00 90 B6 E0 20 D0
.:  0978 EC 86 1C 20 CE F8 00 A2
```

M/C Programming

```
.:   0800 4C 44 F8 00 A2 02 2C A2
.:   0808 00 00 B4 C1 D0 08 B4 C2
.:   0810 D0 02 E6 26 D6 C2 D6 C1
.:   0818 60 20 BB FA 00 C9 20 F0
.:   0820 F9 60 A9 00 00 8D 00 00
.:   0828 01 20 DB FA 00 20 92 FA
.:   0830 00 20 7F FA 00 90 09 60
.:   0838 20 BB FA 00 20 7C FA 00
.:   0840 B0 DE A6 06 9A A9 1C 20
.:   0848 D2 FF A9 3F 20 D2 FF 4C
.:   0850 44 F8 00 20 CB F8 00 CA
.:   0858 D0 FA 60 E6 C3 D0 02 E6
.:   0860 C4 60 A2 02 B5 C0 48 B5
.:   0868 27 95 C0 68 95 27 CA D0
.:   0870 F3 60 A5 28 A4 29 4C 35
.:   0878 FB 00 A5 C3 A4 C4 38 E5
```

```
.:   0A00 90 20 D2 FF 20 26 FD 00
.:   0A08 A9 1E 20 D2 FF 68 20 3C
.:   0A10 FD 00 A2 06 E0 03 D0 12
.:   0A18 A4 1F F0 0E A5 2A C9 E8
.:   0A20 B1 C1 B0 1C 20 C9 FC 00
.:   0A28 88 D0 F2 06 2A 90 0E BD
.:   0A30 2E FF 00 20 A6 FD 00 BD
.:   0A38 34 FF 00 F0 03 20 A6 FD
.:   0A40 00 CA D0 D5 60 20 D4 FC
.:   0A48 00 AA E8 D0 01 C8 98 20
.:   0A50 C9 FC 00 8A 86 1C 20 34
.:   0A58 FA 00 A6 1C 60 A5 1F 38
.:   0A60 A4 C2 AA 10 01 88 65 C1
.:   0A68 90 01 C8 60 A8 4A 90 0B
.:   0A70 4A B0 17 C9 22 F0 13 29
.:   0A78 07 09 80 4A AA BD DD FE
```

```
.:   0A80 00 B0 04 4A 4A 4A 4A 29
.:   0A88 0F D0 04 A0 80 A9 00 00
.:   0A90 AA BD 21 FF 00 85 2A 29
.:   0A98 03 85 1F 98 29 8F AA 98
.:   0AA0 A0 03 E0 8A F0 0B 4A 90
.:   0AA8 08 4A 4A 09 20 88 D0 FA
.:   0AB0 C8 88 D0 F2 60 B1 C1 20
.:   0AB8 C9 FC 00 A2 01 20 0C FB
.:   0AC0 00 C4 1F C8 90 F1 A2 03
.:   0AC8 C4 27 90 F2 60 A8 B9 3B
.:   0AD0 FF 00 85 28 B9 7B FF 00
.:   0AD8 85 29 A9 00 00 A0 05 06
.:   0AE0 29 26 28 2A 88 D0 F8 69
.:   0AE8 3F 20 D2 FF CA D0 EC 4C
.:   0AF0 CB F8 00 20 E3 FA 00 20
.:   0AF8 6C FA 00 20 F4 FA 00 20
```

```
.:   0980 00 00 A0 00 00 B1 C1 DD
.:   0988 10 02 D0 0C C8 E8 E4 1C
.:   0990 D0 F3 20 2D FA 00 20 CB
.:   0998 F8 00 20 D3 F8 00 A6 26
.:   09A0 D0 92 20 31 FB 00 B0 DD
.:   09A8 4C 44 F8 00 20 E3 FA 00
.:   09B0 85 20 A5 C2 85 21 A9 08
.:   09B8 A2 00 00 85 27 86 28 A9
.:   09C0 93 20 D2 FF A9 0B 85 1D
.:   09C8 20 67 FC 00 20 D1 FC 00
.:   09D0 85 C1 84 C2 C6 1D D0 F2
.:   09D8 A9 91 AA 20 43 FA 00 4C
.:   09E0 44 F8 00 A0 2C 20 BB F8
.:   09E8 00 20 CB F8 00 20 2D FA
.:   09F0 00 20 CB F8 00 A2 00 00
.:   09F8 A1 C1 20 E0 FC 00 48 A9
```

```
.:   0B80 C9 0D F0 1E C9 20 F0 F5
.:   0B88 20 D4 FE 00 B0 0F 20 9F
.:   0B90 FA 00 A4 C1 84 C2 85 C1
.:   0B98 A9 30 9D 10 02 E8 9D 10
.:   0BA0 02 E8 D0 DB 86 28 A2 00
.:   0BA8 00 86 26 F0 04 E6 26 F0
.:   0BB0 75 A2 00 00 86 1D A5 26
.:   0BB8 20 E0 FC 00 A6 2A 86 29
.:   0BC0 AA BC 3B FF 00 BD 7B FF
.:   0BC8 00 20 BD FE 00 D0 E3 A2
.:   0BD0 06 E0 03 D0 19 A4 1F F0
.:   0BD8 15 A5 2A C9 E8 A9 30 B0
.:   0BE0 21 20 C3 FE 00 D0 CC 20
.:   0BE8 C5 FE 00 D0 C7 88 D0 EB
.:   0BF0 06 2A 90 0B BC 34 FF 00
.:   0BF8 BD 2E FF 00 20 BD FE 00
```

```
.:   0C00 D0 B5 CA D0 D1 F0 0A 20
.:   0C08 BC FE 00 D0 AB 20 BC FE
.:   0C10 00 D0 A6 A5 28 C5 1D D0
.:   0C18 A0 20 6C FA 00 A4 1F F0
.:   0C20 2B A5 29 C9 9D D0 1D 20
.:   0C28 31 FB 00 90 0A 98 D0 04
.:   0C30 A6 1E 10 0A 4C FC FA 00
.:   0C38 C8 D0 FA A6 1E 10 F6 CA
.:   0C40 CA 8A A4 1F D0 03 B9 C2
.:   0C48 00 00 91 C1 88 D0 F8 A5
.:   0C50 26 91 C1 20 D1 FC 00 85
.:   0C58 C1 84 C2 A0 41 20 BB F8
.:   0C60 00 20 CB F8 00 A9 1E 20
.:   0C68 D2 FF 20 2D FA 00 20 CB
.:   0C70 F8 00 A9 1F 20 D2 FF 4C
.:   0C78 B1 FD 00 A8 20 C3 FE 00
.:   0B00 6C FA 00 A9 09 A2 00 00
.:   0B08 85 27 86 28 20 CE F8 00
.:   0B10 20 6F FC 00 20 D1 FC 00
.:   0B18 85 C1 84 C2 20 70 F7 F0
.:   0B20 05 20 31 FB 00 B0 E9 4C
.:   0B28 44 F8 00 20 E3 FA 00 A9
.:   0B30 03 85 1D 20 BB FA 00 20
.:   0B38 9C F8 00 D0 F8 A5 20 85
.:   0B40 C1 A5 21 85 C2 4C 47 FC
.:   0B48 00 C5 28 F0 03 20 D2 FF
.:   0B50 60 20 E3 FA 00 20 6C FA
.:   0B58 00 8E 11 02 A2 03 20 DB
.:   0B60 FA 00 48 CA D0 F9 A2 03
.:   0B68 68 38 E9 3F A0 05 4A 6E
.:   0B70 11 02 6E 10 02 88 D0 F6
.:   0B78 CA D0 ED A2 02 20 CF FF
```

```
.:   0C80 D0 11 98 F0 0E 86 1C A6
.:   0C88 1D DD 10 02 08 E8 86 1D
.:   0C90 A6 1C 28 60 C9 30 90 03
.:   0C98 C9 47 60 38 60 40 02 45
.:   0CA0 03 D0 08 40 09 30 22 45
.:   0CA8 33 D0 08 40 09 40 02 45
.:   0CB0 33 D0 08 40 09 40 02 45
.:   0CB8 B3 D0 08 40 09 00 00 22
.:   0CC0 44 33 D0 8C 44 00 00 11
.:   0CC8 22 44 33 D0 8C 44 9A 10
.:   0CD0 22 44 33 D0 08 40 09 10
.:   0CD8 22 44 33 D0 08 40 09 62
.:   0CE0 13 78 A9 00 00 21 81 82
.:   0CE8 00 00 00 00 59 4D 91 92
.:   0CF0 86 4A 85 9D 2C 29 2C 23
.:   0CF8 28 24 59 00 00 58 24 24
```

```
.:   0D00 00 00 1C 8A 1C 23 5D 8B
.:   0D08 1B A1 9D 8A 1D 23 9D 8B
.:   0D10 1D A1 00 00 29 19 AE 69
.:   0D18 A8 19 23 24 53 1B 23 24
.:   0D20 53 19 A1 00 00 1A 5B 5B
.:   0D28 A5 69 24 24 AE AE A8 AD
.:   0D30 29 00 00 7C 00 00 15 9C
.:   0D38 6D 9C A5 69 29 53 84 13
.:   0D40 34 11 A5 69 23 A0 D8 62
.:   0D48 5A 48 26 62 94 88 54 44
.:   0D50 C8 54 68 44 E8 94 00 00
.:   0D58 B4 08 84 74 B4 28 6E 74
.:   0D60 F4 CC 4A 72 F2 A4 8A 00
.:   0D68 00 AA A2 A2 74 74 74 72
.:   0D70 44 68 B2 32 B2 00 00 22
.:   0D78 00 00 1A 1A 26 26 72 72
```

```
.:   0D80 88 C8 C4 CA 26 48 44 44
.:   0D88 A2 C8 3A 3B 52 4D 47 58
.:   0D90 4C 53 54 46 48 44 50 2C
.:   0D98 41 4C F9 00 3F F9 00 DD
.:   0DA0 F8 00 06 F9 00 60 F9 00
.:   0DA8 87 F9 00 E9 F9 00 FD F9
.:   0DB0 00 40 FB 00 94 FB 00 C2
.:   0DB8 FB 00 35 FC 00 5D FD 00
.:   0DC0 8B FD 00 AD FD 00 17 F8
.:   0DC8 00 0D 20 20 20 50 43 20
.:   0DD0 20 53 52 20 41 43 20 58
.:   0DD8 52 20 59 52 20 53 50 AA
.:   0DE0 AA AA AA AA AA AA AA AA
.:   0DE8 AA AA AA AA AA AA AA AA
.:   0DF0 AA AA AA AA AA AA AA AA
.:   0DF8 AA AA AA AA AA AA AA AA
```

```
T=0:FORJ=1024TO3550:T=T+PEEK(J):NEXT:?T
 283370

READY.
T=0:FORJ=1024TO1663:T=T+PEEK(J):NEXT:?T
 68631

READY.
T=0:FORJ=1664TO2303:T=T+PEEK(J):NEXT:?T
 77155

READY.
T=0:FORJ=2304TO2943:T=T+PEEK(J):NEXT:?T
 74768

READY.
T=0:FORJ=2944TO3550:T=T+PEEK(J):NEXT:?T
 62816

READY.
```

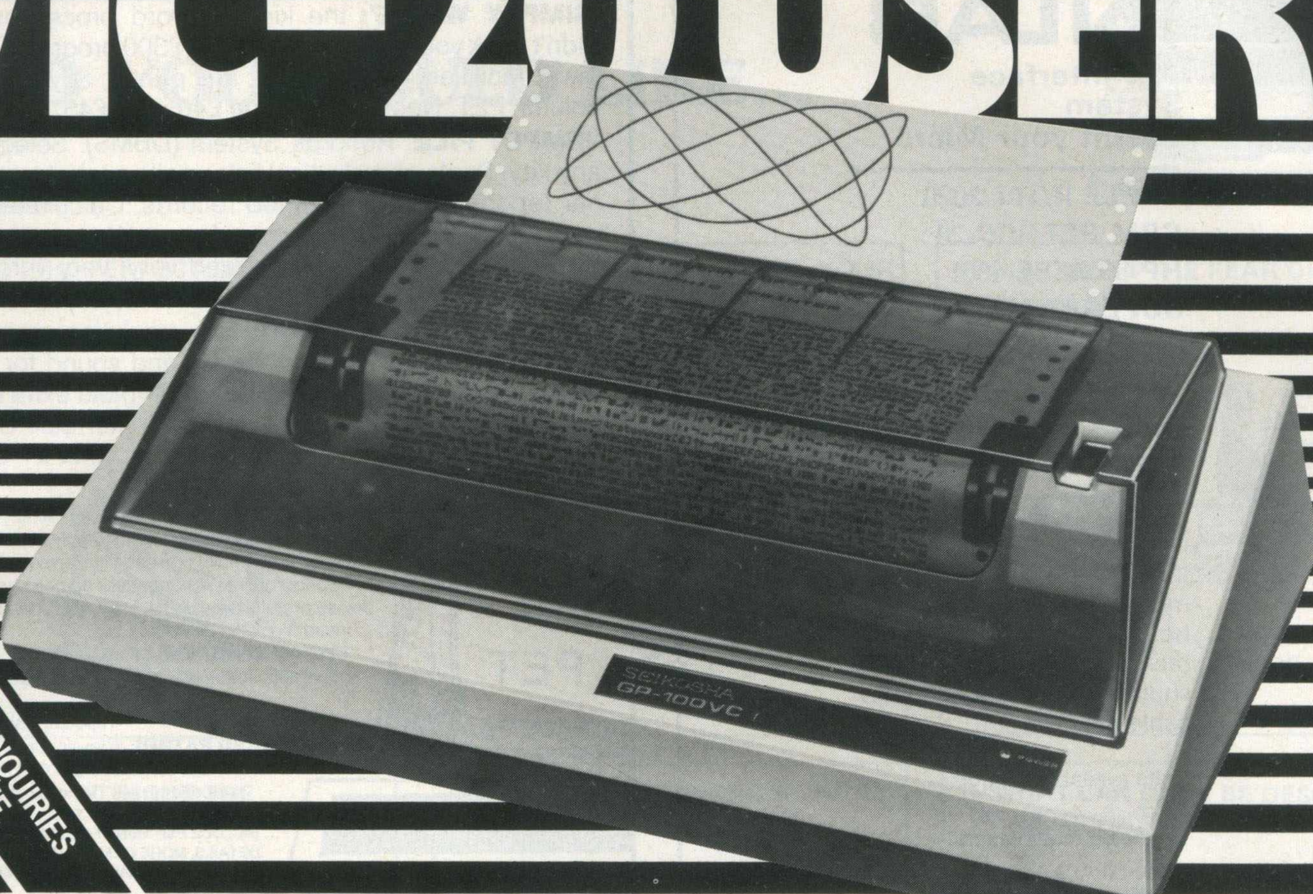
```
FORJ=1024TO1663STEP8:T=0:FORK=JTOJ+7
:T=T +PEEK(K):NEXT:?T,:NEXT

 464       382       565       426
 474       451       472       587
 447       538       579       481
 655       774       510       850
 756       780       802       910
 886       853       801       784
 876       840       835       1383
 753       1190      1134      831
 816       803       753       883
 686       889       850       893
 593       987       1415      1035
 1088      1079      1070      762
 1132      853       825       1193
 639       1207      983       824
 975       1041      744       1408
 965       1082      1230      1139
 664       793       768       987
 905       617       1049      1218
 1192      794       1201      803
 858       1030      805       1036
```

```
FORJ=1664TO2303STEP8:T=0:FORK=JTOJ+7
:T=T +PEEK(K):NEXT:?T,:NEXT

 1196      1068      713       682
 646       683       913       1191
 1045      1417      835       1166
 1417      1224      1150      951
 1232      1143      991       953
 884       1103      796       1284
 1274      937       1277      835
 902       770       638       1227
 876       912       714       660
 1133      1388      894       893
 659       919       825       500
 854       708       443       1340
 762       963       1293      1038
 655       930       658       875
 953       1270      1001      1419
 1082      1082      878       1256
 1104      775       834       858
 806       837       968       1185
 645       1301      782       1003
 943       963       916       859
```

```
FORJ=2304TO2943STEP8:T=0:FORK=JTOJ+7
:T=T +PEEK(K):NEXT:?T,:NEXT

 835       672       923       1265
 901       916       859       881
 937       901       1193      589
 999       989       1179      1046
 751       926       1013      943
 1083      901       867       677
 986       880       1329      909
 987       810       645       1103
 964       892       874       1089
 1059      981       941       1001
 1215      995       837       792
 975       838       808       1052
 517       684       863       825
 994       791       1284      911
 977       1129      991       514
 858       1282      992       916
 698       832       888       1283
 822       994       666       1190
 1117      977       995       577
 1146      805       737       1305
```

```
FORJ=2944TO3551STEP8:T=0:FORK=JTOJ+7
:T=T +PEEK(K):NEXT:?T,:NEXT

 1202      880       1259      797
 997       924       645       891
 1239      1072      901       1054
 958       1437      698       965
 1290      1295      1013      985
 876       818       800       1318
 1125      1095      1002      1211
 714       1027      1021      1079
 959       671       726       655
 443       475       475       502
 552       739       458       540
 600       457       662       325
 461       843       535       445
 477       1031      342       810
 848       734       836       882
 1180      956       612       356
 980       797       577       923
 848       1368      908       902
 1089      288       481       564
```

VIC-20 USERS



DEALER ENQUIRIES WELCOME.

The new Seikosha GP-100VC graphics printer for around £235. EX. VAT.

Offering big printer performance at a fraction of the cost, the latest addition to the famous range of Seikosha micro-printers is the 100VC. The precise match for the VIC 20.

Featuring all the VIC 20 characters, symbols and graphics as standard, the Seikosha 100VC includes full graphics capability. It enables graphic, character and double width character modes to be intermixed on a single line as well as repeating graphics data, as you want, with a single command.

Many other advanced features, plus Seikosha's proven reliability and the nationwide support of DRG's distributor network make the 100VC the natural choice for the VC10 user.

DIMENSIONS:

Depth – 9¼″ (234mm)

Width – 17¼″ (420mm)

Height – 5¼″ (136mm)

OPTIONS:

Interfacing for most other systems available on the GP100A model.

FEATURES INCLUDE:

- 80 col. 30 cps.
- Dot Matrix unihammer action.
- 154 characters (inc graphics)
- VIC-20 8-BIT CODE
- Full graphics.
- Double width printing.
- Automatic printing
- Up to 10″ paper width.
- Original + 2 copies.
- Tractor feed.
- Self testing.

DRG
BUSINESS
MACHINES

Telephone the number below and we'll tell you where your nearest distributor is located. See the remarkable Seikosha GP100VC in action

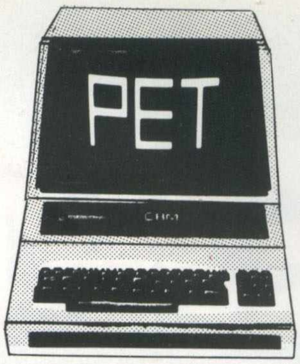
(Peripherals & Supplies Division) 13/14 Lynx Crescent, Winterstoke Road,
Weston-super-Mare, BS24 9DN. Tel: (0934) 416392.

THE FINEST WORLDWIDE SUPPORTED NATIONWIDE.

DRG (UK) Ltd, Reg No. 22419 England.

ADVERTISER'S INDEX

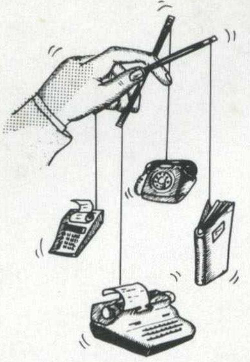
IF YOU HAVE ONE OF THESE...



AND YOU WOULD LIKE...

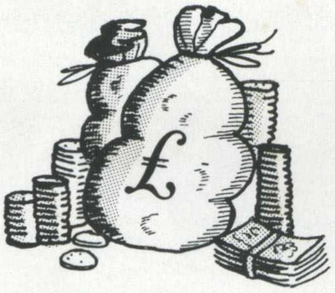
THE PERFECTION OF A TAILORED SOLUTION

THE ABILITY TO AUTOMATE YOUR BUSINESS ROUTINES THE WAY YOU WANT



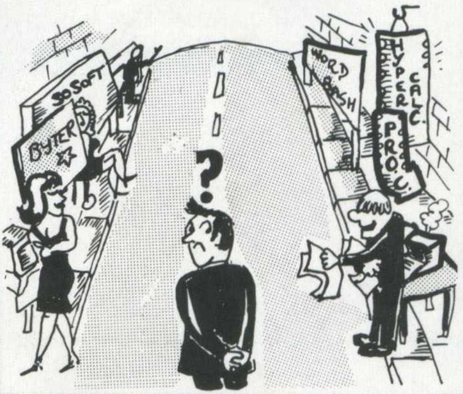
WITHOUT THE FEAR OF MAKING THE WRONG DECISION

WITHOUT THE HIGH COST



WITHOUT BEING A COMPUTER EXPERT

IN WHAT CAN ONLY BE DESCRIBED AS A CONFUSING MARKET



SILICON OFFICE DOES IT ALL

Add muscle to your 8032 with a memory expansion board and SILICON OFFICE and give yourself some elbow room. SILICON OFFICE is a leading software product for the Commodore 8096, that offers you the ability to create and control your application system(s) the way you want it.

The unique 'data base' facility in SILICON OFFICE has easy to use routines to draw record cards on the screen, insert and 'housekeep' your file information and then combine and analyse data from several files to produce complex reports designed to your needs. Simple but repetitive routines can be stored for subsequent execution. A powerful word processor is built in, available at all times, with the emphasis on ease of use for the average typist.

In fact SILICON OFFICE can be considered as a product offering three packages in one with communications complementing the data base and word processor. If you already own a Commodore 8032 this can be quickly upgraded to the latest technology micro-computer – add SILICON OFFICE and you will have a total and cost effective solution to your software needs for years to come and all of this for just over £1000. For more details contact your Commodore dealer or complete the coupon below.

PLEASE SEND ME MORE DETAILS ON SILICON OFFICE

BSF

Name ____________________

Position ____________________

Company ____________________

Address ____________________ Telephone: ____________________

Send to: Bristol Software Factory, Kingsons House, Grove Avenue, Queen Square, Bristol BS1 4QY. Telephone 0272 277135

STACK
ANNOUNCE THE NEW COMMODORE 64

- 64K of RAM....20K of internal ROM
- Superb colour graphics and sound
- Unbelievable SPRITE graphics
- 40 columns
- 16 colours
- Programable attack, decay, sustain, pitch, wave form, volume....practically a music synthesizer!
- CP/M capability as a plug-in option
- TV output

STACK also have available for the CBM 64...LIGHTPEN, RS232 INTERFACES, ANALOGUE JOYSTICK for SPRITE motion, MEMORY EXPANSION....

ORDER YOUR COMMODORE 64 NOW FROM STACK TO ENSURE EARLY DELIVERY

£299 plus VAT
(shipping £6 plus VAT)

IDEAL FOR EDUCATION

MULTI·BUS (IEEE)

for Pet Multi-Users

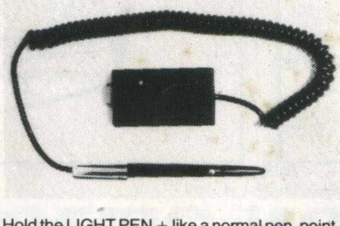
- A single ROM for each Pet* in the system uses the Pets' own intelligences to organise the sharing of any CBM disc drive.
- No external hardware required.
- Uses standard Pet cabling.
- Works with most standard software including Superscript, Comal, Pascal, etc..
- Compatible with TURBO ROM.
- Many different files of the same or different types may be processed simultaneously or even shared.
- A screen message indicates if delays are encountered and original screen contents are restored entirely when bus is free.

* for 4000, 8000 series, any Basic 4 machine including upgrades.

A Neat System at a Neat Price!

Starter Set (2 Pets)	£81.00+ VAT
Extra Pets (ordered with above)	£27.00+ VAT
Later Additions	£36.00+ VAT
Special ROM required for 9'' Pets	£19.00+ VAT

VIC·20 LIGHT PEN + NEW LIGHT PEN GAMES



Hold the LIGHT PEN + like a normal pen, point it at your TV screen and the pen tells your VIC-20 what it sees! Instead of pressing keys, touch the screen with your LIGHT PEN + and move your man in CHESS, create LIFE, play OTHELLO, play GO. Point the pen at your man and move him to his new position. Many games are already available for the LIGHT PEN + and many more are being developed. You can write progams which allow you to read with your LIGHT PEN + from the TV screen or place images or words on the screen. e.g. From a long list on your screen use the LIGHT PEN + to choose the facts and figures you want. The well-known game CONCENTRATION is supplied on cassette tape FREE with every LIGHT PEN +.

only **£25.00** plus VAT
FREE GAME SUPPLIED!!

Light Pen Go (colour and sound)
The program allows two players to play this traditional and fascinating Oriental game placing and removing pieces using the LIGHT PEN +.

Light Pen Othello (colour and sound)
A two-player game placing and reversing pieces in this high strategy game using the LIGHT PEN +.

Light Pen Draughts (colour and sound)
The program allows two players to play against each other using the pen to move and take, including kings and blocking of illegal moves. Requires a 3K RAM pack.

Light Pen Concentration (colour and sound)
A two-player game in which a pack of cards are shown face down on the screen and turned over by the LIGHT PEN + in a search for pairs. Highest score wins.

Light Pen Life (colour and sound)
In this cult game pieces are placed in the universe and grow or die like bacteria colonies in life giving fascinating gliding or pulsing displays.

Cassette based games for use with STACKS' Commodore Approved LIGHT PEN +

only **£5.00** each plus VAT

ANALOGUE JOYSTICK

only **£13.00** plus VAT

Programming the VIC-20? Why not write your very own games! The ANALOGUE JOYSTICK will add another dimension to the games you write. It gives you full Joystick control in ANY direction — and it fires too! Move your spaceship in a figure of eight or loop the loop — you are not restricted to up, down, right, left any longer!

Contact your local Commodore VIC dealer for details.

Stack Computer Services Limited, 290-298 Derby Road, Bootle, Merseyside. 051-933 5511. Telex: 627026.