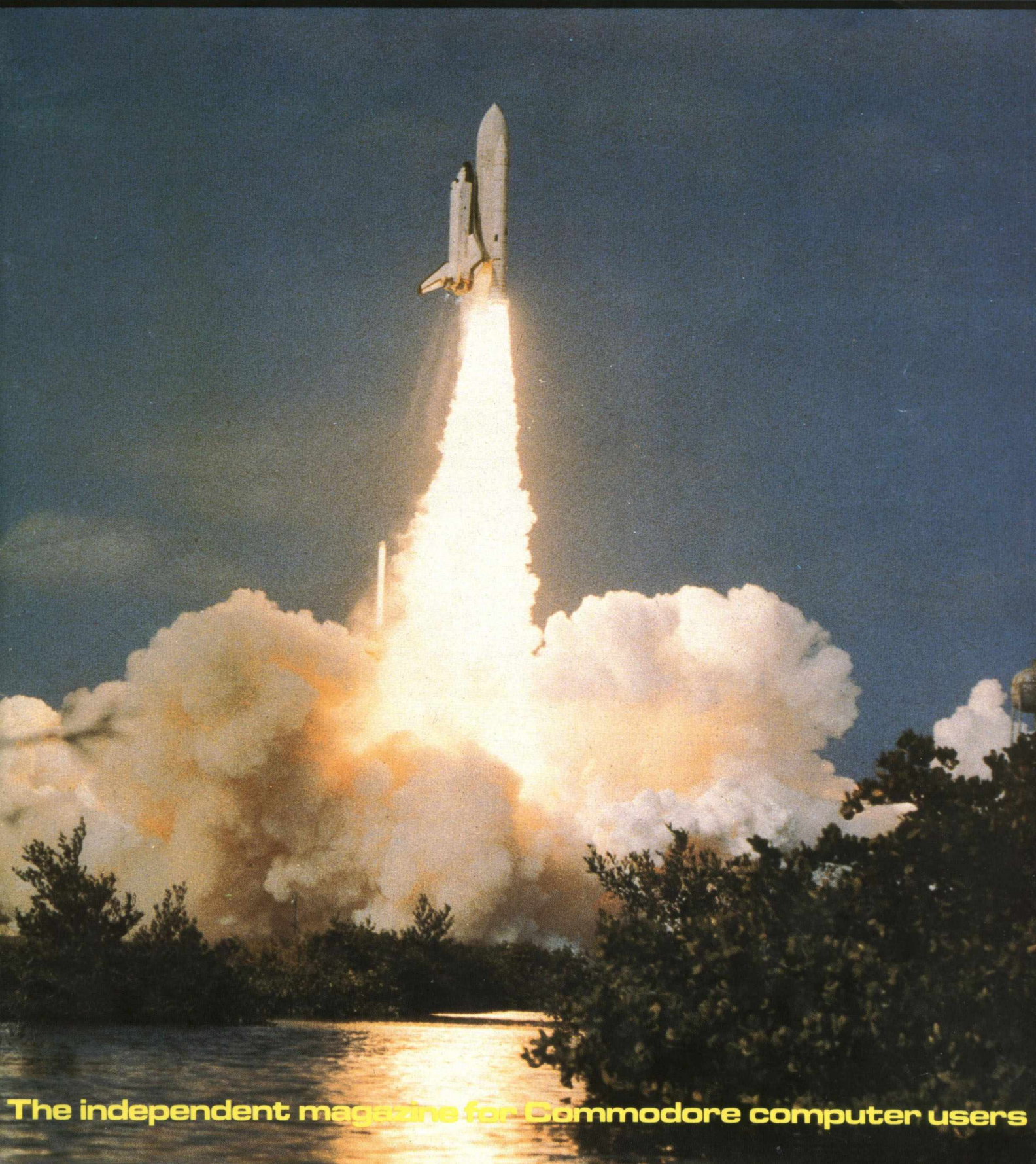
commodore

COMPUTING

international

September 1982 £1.00



The independent magazine for Commodore computer users

VICAID

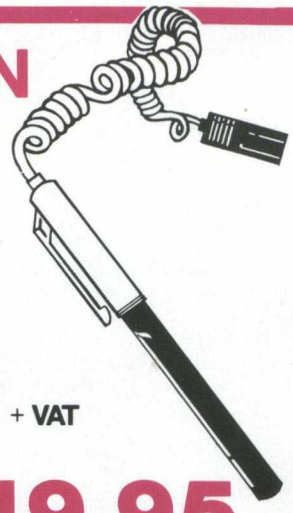
Two sets of Fabulous Utilities in one!

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

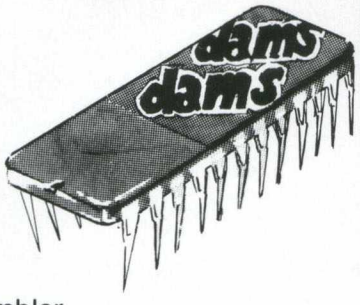
DAMS PRICE ONLY

£17.35 + VAT

FOR PET 12" SCREEN **£19.95** + VAT

VICMON

THE ULTIMATE PROGRAMING AID FOR THE VIC

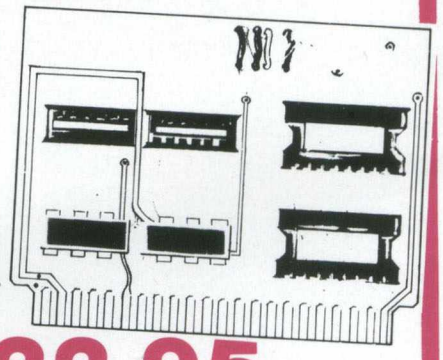


Full machine code package with:
Assembler, Disassembler,
Fill, Re-locate, Identify, Exchange, Compare, Printing, Dissassembler etc., etc.

Needs DAMS RAM/ROM board or similar

£19.95 + VAT

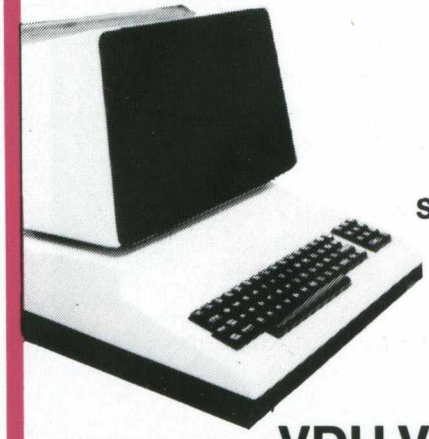
RAM 'N ROM BOARD



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

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

Editorial Assistant
Fiona McCormick

Production
Three's Company

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 Edwin Snell printers, Yeovil, England.

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

Commodore Computing
193 Wardour Street
London WC2

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 the best of our knowledge, there are some half a dozen specialist Commodore magazines around the world, five of which are written in the English language. Of these, two come from America, and are produced by the company themselves. Power/Play, a quarterly, is aimed very much at the Vic end of the market, and concentrates (as its name suggests) very heavily on the games side of Vic.

Commodore The Microcomputer Magazine is a bi-monthly publication, covering the whole range of Commodore equipment, but to be quite frank the level of content over the entire magazine is quite low, with only a couple of articles worthy of serious technical consideration.

The Australian magazine is again an in-house publication, which seems to rely very heavily on all the rest of the magazines for its actual content, although to be fair it does spread itself over the complete spectrum of Pets and Vics available.

The fifth one, apart from ourselves of course, is a Canadian publication called The Transactor, recently taken over by Canadian Micro Distributors: it too used to be in-house. This one is to be taken very seriously, and has an extremely good technical content level. You may have seen some material from Transactor reproduced here and elsewhere.

This brings me to the heart of the matter: reproduction of material from other Commodore magazines. I am not averse to doing this, because although we get issued magazines free of charge, the vast majority of you will not even be seeing them. As some of the material is so good, it is only fair to bring it to your attention.

Obviously we will limit the amount of duplicated material, but where it's of sufficient interest it is useful for as many people to see it as possible. People keep re-inventing the wheel: there's no reason for you to have to do so.

Letters

Dear Sirs,

It seems that there is a distinct lack of programs in certain areas as far as Commodore Pets are concerned. I wonder if you can assist further by recommending a good source of available software — for instructional learning purposes — games, and if possible technical applications in architecture/building. If you could assist me I would be very grateful.

Yours faithfully
J. Randall
Woking

Dear Mr. Randall,

There are a number of suppliers of the kind of software you require. On the instructional learning side, probably your best bet would be a company called ESM, who produce a wide range of such software. They can be reached on 0945-63444.

Games, well everyone's producing games at the moment. Two companies it would be worth calling are Audiogenic (0734-586334) and Supersoft (01-861 1166), both of whom produce a large number of games for the Pet.

Technical Applications in architecture/building are obviously a different kettle of fish, but three companies to contact, all of which produce software in this area, are Claremont Controls (0669-21081), Computer and Design Services Ltd. (0202-697341) and the Technical Software Centre (0234-750102).

We hope the above is of assistance to you.

Dear Sirs,

I am a newcomer to computing, having recently purchased a Vic 20. Any information on learning and reference books would be appreciated.

Thanking you in anticipation.

Yours faithfully
Mrs. Bunney
Fareham

Dear Mrs. Bunney,

The Vic 20 'suffers' from an almost overwhelming number of books and instructional aids at present, some of them good, some of them dreadful. Surprisingly, in view of their track record in this area, a couple of the best items about at the moment come from Commodore. Their book the Vic Programmers Reference guide deserves an investigation, as it covers an awful lot of ground from assuming no prior knowledge of the machine.

On the software front, and aimed more at the beginner, is An Introduction To Basic part 1, with the promise of part 2 to come sometime. This consists of a cassette and a voluminous manual, which is extremely well written. Both of these should be available from any Commodore dealer.

On a more general front, you could do worse than go into your local W.H. Smiths and have a browse in their computer books section. They sell quite a few now, and one of them should attract your attention.

Dear Sirs,

Please find enclosed remittance for 12.50 pounds, for one years subscription to Commodore Computing.

If you have run an article previously on how to assign the function keys on the Vic 20, could you please forward a copy of that article if possible.

Thanking You
Thomas Gallivan
Tralee

Dear Thomas,

Thanks for the cheque: magazines will be winging their way to you!

We haven't published an article on assigning function keys, but here goes.

Basically, there is not a lot that can be done on the standard Vic. Before serious assignment can start taking place you're going to need something like the Super Expander or another analogous cartridge. However, we can detect which one is being pressed, and by seeing if the Commodore logo key or the shift key is pressed as well, we have a number of different options available to us.

Memory location 203 returns a certain value if the shift key is placed, and location 653 has the following values:— 0 if shift not pressed, 1 if pressed, 2 if the logo key is pressed, and 3 if both the shift and the logo key are pressed.

The following program will tell you which key is pressed.

```
10  A=PEEK (203)
15  B=PEEK (653)
20  K=0
25  IF A=39 THEN
K=1:GOTO 50
30  IF A=47 THEN
K=3:GOTO 50
35  IF A=55 THEN
K=5:GOTO 50
40  IF A=63 THEN
K=7:GOTO 50
45  GOTO 10
50  IF B    1 THEN B=0
55  K=K+B
60  PRINT ''FUNCTION KEY''
'';K;'' PRESSED''
65  GOTO 10
```

Dear Sirs,

I have two 32K Pets series 2000 and 3000, interfaced to Centronics 701 and 703 printers respectively, the former at home for programming purposes — my interest being exclusively business.

All my programmes are stand alone, being cassette based, which can be a nuisance at times, where for instance a debtor changes his address. With backup tapes this can take an hour or more to save and verify.

I have held my hand as to discs — especially as Commodore don't enjoy a good reputation in some quarters here — for I want to be able to access any part at random within five seconds at most.

Can you advise on hard disks?

Yours faithfully
William Nesbitt
Foxrock

Dear Mr. Nesbitt,

I can quite understand you desire to become disk-based: when you eventually take the plunge you'll wonder how you ever managed without!

I don't really share your concern over the Commodore disk drives: I've been using an 8050 for a long time now, and no discernable problems have arisen as yet. You say you require an access time of 'at most' five seconds. Bearing in mind that this is the average delay time for a CLOSE and OPEN file loop, you should be able to get at any record in less than a second.

The major hard disk unit on the market at present comes from Mator Systems Ltd. They do a 22 Megabyte unit, at 3,695 pounds, a 30 megabyte unit at 4,095 pounds, and a tape backup system (which takes approximately 45 minutes to backup a 22 meg. system) at 1,895 pounds.

Personally, I feel an 8050 would be adequate for your purposes, but if you want to contact Mator, their number is 0273-720451.

Dear Sirs,

Please could you advise on whether there is a simple way of preventing a program on a PET 4032 from being listed or amended by the operator.

We have an application at work where this is necessary. With a CP/M based machine the instruction 'SAVE . . .P'' will prevent the program from being listed when loaded.

Yours faithfully
Alan Mason
Leicester

Dear Alan,

Unfortunately there is no simple way to prevent a program being listed or amended. Any method that you try and implement can usually be got around very easily, so we can't really come up with an answer. If any readers have any bright ideas, we'd be delighted to hear from you.

Meanwhile, two suggestions to be going on with. One is to buy Raeto West's Programming the Pet/CBM, which has a couple of pages devoted to discussions on a Basic command which he calls UNLIST: this performs precisely the function you're interested in, were it possible to perform the command satisfactorily.

The other is to buy one of the number of compilers around at the moment, and present your operator with a compiled version of the program. This, when listed, will simply display one line number with one SYS call: unlistable and unalterable.

New Product News

So many companies are producing books on the Pet and/or Vic these days that it's almost impossible to know where to start. Still, there are a number of major distributors who have made new contributions lately, so perhaps we can point you in the right direction with those.

John Wiley and Sons Ltd.

Their latest catalogue covers a whole host of general books on computing, but have quite a collection of specific Commodore interest. Again, these are mainly introductory guides to programming in Basic, but many other sections are covered as well.

They have also a large selection of software items as well, including a version of the famous Sargon chess program, a guide to general mathematics, a book on Pet-aided design of active filters, and so on. Worth checking out, and they can be reached on 0243-784531.

Osborne/McGraw-Hill

They are in really the same position as John Wiley, in that they have an awful lot of books on general computing topics, and a variety of specifics. Worthy of mention is the Hands-on-Basic with a Pet, an introduction to Basic programming on the Pet (and hence the Vic as well), which is a very well put-together book, and extremely useful for the beginner to programming.

Amongst the others is a Fun and Games book, a collection of programs previously published (or should I say taped) by the well-known cassette magazine Cursor. It makes a change from the usual run of the mill games book in that all the programs are superbly legible (and what's more they work!). We take a more detailed look at this one in our book reviews section this month.

Many others are done by Osborne/McGraw-Hill. Possibly the best one is the Pet/CBM Personal Computer Guide, written by Adam Osborne with Carroll Donahue. For a while Commodore themselves where shipping out this book with every Pet delivered, so highly did they think of it. A very comprehensive guide for the newcomer to the Pet, convering all the major peripherals on the way.

Further information on Osborne/McGraw-Hill can be obtained by ringing 0628-23431.

Prentice-Hall and Gower Press

These last two companies are similar to our first two, in that amongst the general plethora of books they publish, there are a number devoted to the Pet and/or Vic. Both are worth giving a ring: Prentice-Hall are on 0442-58531, and Gower are on 0252-331551.

Other Suppliers

Of course, these are not the only book publishers currently producing books on the Commodore range of equipment. If any of you others are out there, drop the editor a line, and we'll take it from there.

Prestel

A lot has been written in the computer press and elsewhere about the rapidly expanding world of Prestel, unfortunately usually accompanied by the rapidly expanding cost of Prestel. A company down in Bristol (Avon Office Services, on 0272-502008) have now produced a low-cost adaptor for any television and microcomputer, which brings access to Prestel that much easier and cheaper.

Known as Tandata, this product will interface to the Pet, allowing it to store, review and process Prestel and other viewdata pages on and off-line. Plus, of course, you also have access to the standard Prestel character set, giving you an 80×24 display, 8 colour display, with (amongst other wonders) double height graphic characters.

INSTALLATION SERVICE

Experience is often very valuable when installing your new Commodore System. Mistakes are frequently very costly and waste valuable time.

ONE DAY SERVICE
For a fee of £85 per day plus expenses a member of staff will help you overcome early difficulties and set you on a suitable path to a successful computerisation.

FULL INSTALLATION SERVICE
This is tailored to your requirements. We can supply extra operations staff, or technical advisors. Extra equipment can be useful if an installation is required by a certain date. As full Commodore Systems Distributors we have experience you will probably need.

MAINTENANCE
Most of the system breakdowns are not hardware faults, but consist of lack of understanding of programs or faults based upon unwise practices. Our staff are trained to assist with system problems, and they are capable of finding the best possible solutions. Maintenance staff will visit your site, diagnose your difficulty and if necessary replace any components needed.

For information concerning this service please contact Brian Homewood or Robert Jones.

PEACH DATA SERVICES LTD.

COMPUTER SERVICE TO BUSINESS

5 HORNINGLOW STREET, BURTON-ON-TRENT, STAFFS. BURTON (0283) 44968

New Product News

The unit comes complete with all the necessary gumph to connect it up, but British Telecom make their presence felt making it essential that you contact them for installation of a Prestel jack socket.

Very Important Products

A team of people down in the new 'area to be in' Milton Keynes, have come up with a tremendous graphics display system, which like the aforementioned Prestel adaptor will work with any microcomputer currently available, although of course our interest centres on the Pet.

Known as Micro Design (although they like to call themselves The Solution People), the VIP system connects your microcomputer up to a television set, to give you a resolution display area of 512 by 512 pixels. On screen at any time you can have up to 15 colours selected from a 'palette' of 4,096 different colours (bet you didn't even know there were than many!).

Using the Picaso software package, you have ready access to over 500 subroutines for extremely powerful and fast 2 or 3 dimensional picture manipulation. The accompanying illustrations should give you some idea of the capabilities given to you: I'm only sorry they're just in black and white.

Some of the Picasso commands are very interesting. For example, 'Explod', which displays the individual elements of a picture gradually moving apart. There are also a large number of rotational algorithm provided as well, turning your Pet into a picture-based word processor of tremendous power.

Already at home in a number of scientific and industrial area, VIP is quite a revolution in terms of graphics capabilities. The cost of VIP itself is around 2,900 pounds, and by the time you've done all the IEEE interfacing and all the other connecting up necessary, the complete system should still cost you well under 10,000 pounds. For further details contact 0908 663655.

Like Blood to a Vampire

3D Digital Design and Development have developed a combined hardware and software package to help make routine blood analysis a lot easier.

Using the tried and trusted Coulter Blood Analyser linked up to a Pet, the system handles all the administration, label writing, result and statistical checking necessary in work of this sort.

Along with the relatively low-cost Commodore dot matrix printer, the whole system is very good value for all the tedium it takes out of this normally boring task. If you're interested, your contact is Dr. Mills, on 01-387 7388.

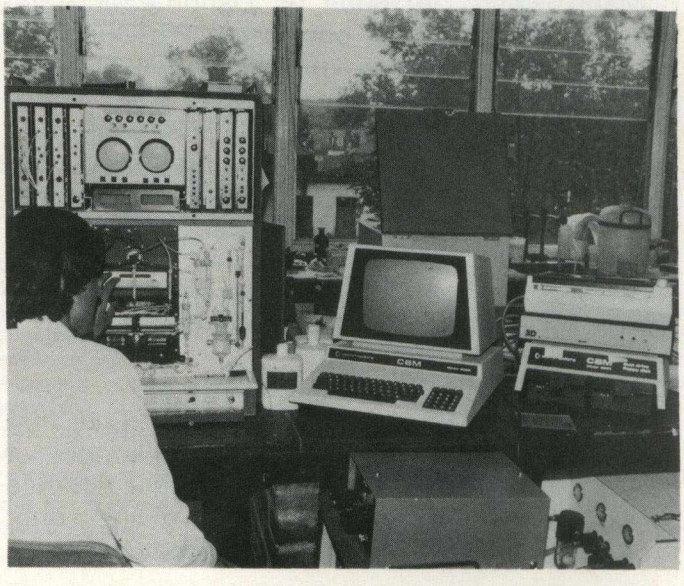
Flexirom

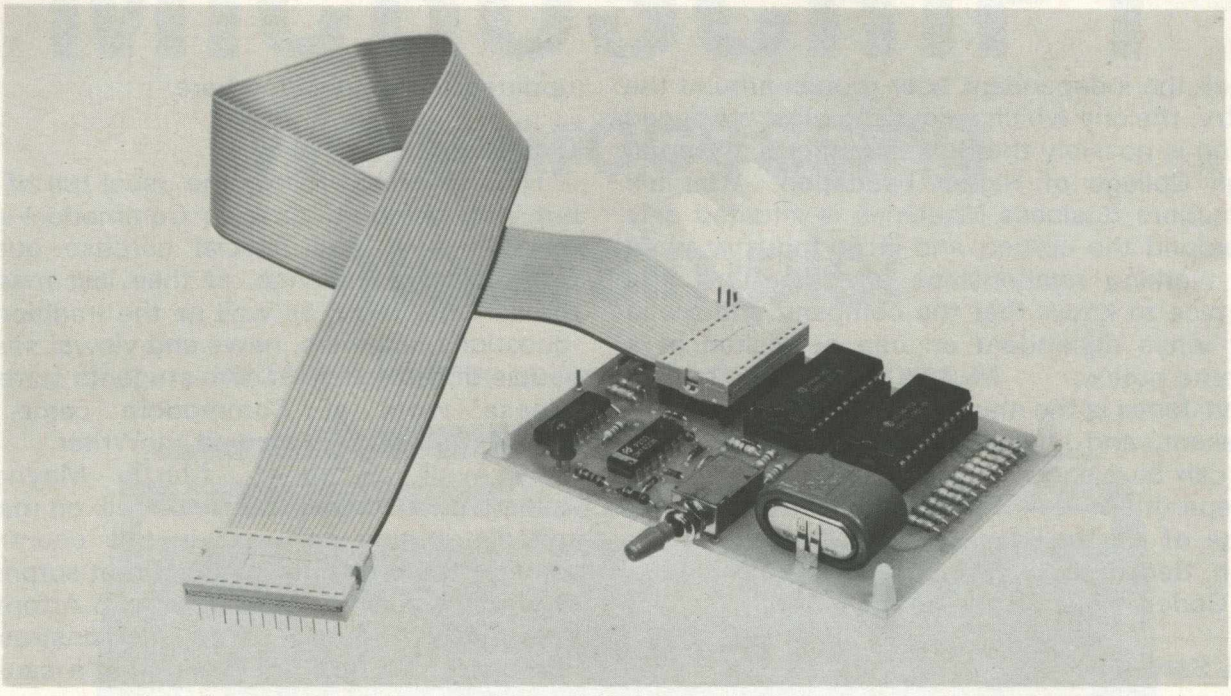
As we all know, all that RAM in your Pet or Vic is lost when the machine is powered off. Even the usual alternative EPROMs, which can quite happily maintain all the information when turned off, require erasing and re-programming by a (usually) expensive Eprom programmer, even if just one byte is being modified. So, Clever Computer Controls (who thinks of these names?) have come up with the answer.

Calling their system Flexirom, it is essentially programmed like normal RAM, but can be switched to ROM mode so as to retain data when the power is removed. It uses a 4K CMOS RAM board, complete with Ni-Cad battery, and is supplied with a 24 way DIP to DIP ribbon cable which connects into any of the usual spare sockets in the internals of your machine.

A simple 'at a stroke' operation converts all your coded RAM into ROM, and this ROM can be subsequently very easily altered without having to resort to the old Eprom programmer. At only 39.50 pounds, this is a veritable bargain.

For more information ring 022 770 644.





From Floppy to Hard

Mator Systems Ltd. (0273-720451) have just released an advanced version of their hard disk unit for the Pet. The Shark, as it's known, has a specially developed 'Chichester' controller, enabling access of the high storage capacities of Winchester disks. Using hard disks, Mator now have a 22 megabyte and a 30 megabyte disk system available.

It operates as far as the Pet is concerned as a 'souped-up' 8050, using all the usual commands. Thus, programs written for the 8050 will work on these new drives, provided all the relevant data is copied over of course.

Unlike the earlier Commodore disk units, you do not require extra add-ons to work in a multi-station environment: they can be simply chained together.

Also available from Mator is a backup system for the hard disk, known as the Sharkive. This is a tape streaming device, incorporating 300XL and 600A quarter inch removable magnetic tape cartridges. Connecting directly up to the Shark drive, a backup of a full 22 megabyte disk takes approximately 45 minutes, including full read-write checking.

Pricing starts at 1,895 for the backup unit, 3,695 for the 22 meg. device and 4,095 for the 30 meg. one.

Compared to the price of the existing Commodore 1 megabyte disk unit, these prices are relatively low, and deserve looking into by anyone with a serious computing requirement.

Testing Technology

Usually the most testing technology comes from reading all the blurb that is sent in as press releases, but at least Engineering Laboratory Equipment Ltd. (0442-50221) have some useful and interesting products to announce. All to do with monitoring scientific and engineering equipment, the heart of the matter is something called Dataface System 3, which links the Pet to the outside world in a 'unique' manner.

Space forbids me expanding on this one, but if you're interested I would certainly give them a call for further information. They also have a range of programs available to consolidate the Dataface: direct shearing, transducer calibration are just two of the ones available.

Superscript

We mentioned in last months Club News section, the story concerning Superscript, the £35 Wordprocessor from ICPUG. At the time there was a court case pending, which meant that temporarily Superscript was taken off the market.

However, all appears now to have been resolved and the company called Precision Software are now marketing the product in the U.K. A Mr Turner, on 01-330 7166, has all the details. At present they are talking about a selling price of around £240: if you read the Club News section last month you will know what my opinion of this is!

It is interesting to note that Commodore have taken on the rights for Superscript for all of their new machines (Commodore 64, etc.), I wonder what difference colour and sound will make to a Wordprocessor!

Next month we will be bringing you a detailed review of this package.

Club News

Of all the independent user groups around the country, the one which enjoys the most privileged position is possibly the Berkshire group, based at Slough College of Higher Education. After all, Commodore Business Machines is situated only just around the corner, and in an industry were good working relationships are important, it's quite nice to know that the company you are in many ways dependant on can be visited at a moments notice.

Brian Jones is the man in charge of affairs there at present, and any further information you require can be gleaned by either writing to him at the Department of Maths and Computing, Slough College of Higher Education, Wellington Street, Slough, Berkshire, or giving him a ring on Slough (STD Code 0753) 34585 extension 81.

Locale Served

As you might ascertain from the name of the group, they cover the whole of the Berkshire area. One wonders if the future King of England ever pops in to visit them on user nights? After all, Buckingham Palace does possess at least one Commodore system!

So basically, anyone who is in and around the Slough area is more than welcome to come along to the meetings, details of which can be gleaned from Brian on the aforementioned number. For that matter, one assumes that they wouldn't be too offended if people came from further afield as well.

Potted History

The group was the first new one to be founded after the Vic was beginning to get off the ground in this country, and thus they have a fair sprinkling of Vic users in the group. An estimate puts the figure at around 25%-30%, although interestingly enough it's probably nearer to a 50/50 split on actual club nights.

There are no plans at present to form a separate Vic users group, as the interests of all seem to be well enough served as the set-up stands at the moment. Long term of course this may well change, as Vic and its derivatives become more and more well established, but we can only wait and see.

Meetings of the group are generally held on a monthly basis during term times: outside of this, there are not meetings going on. The location is usually the college itself, although they have spread their wings as far afield as their sister college at Langley, Crane Packing on the Slough Trading Estate, and are currently looking for a more permanent meeting place. We'll keep you updated on the position here.

Club Nights

Their meetings follow the usual run of things, but with being so close to Commodore they do tend to have the special surprise guest appearance. For instance, at their last meeting of the summer term, as well as the traditional club 'questions, answers, news and views' session, a couple of the college's own students currently on release work at Commodore came in to demonstrate Simplicalc and VicWriter.

As well as those, Martin Maynard of Audiogenic popped in to give a talk on marketing and distributing software in this country, and some of the problems involved (not surprising, in view of his recent encounters with Arfon). Also, they were treated to a demonstration of PetSpeed, the basic compiler that's causing so much fuss at the moment.

Recently, John Collins of Commodore came along to chat about the new machines coming up later this year and early next, and other people from the company have come along from time to time as well.

This 'privileged position' was shown well a couple of months ago when, just after the Pet Show, they were able to borrow a copy of the promotional video Commodore were using to show off the new equipment, so that all the people who couldn't get to the Show, or who missed it when they were there, had a chance to at least see the machines.

Other Activities

Aside from user group activities, the college also gets involved in other projects as well. Held at the college recently was the South East Study Group for Teachers in Further Education, which featured a number of demonstrations on Commodore equipment.

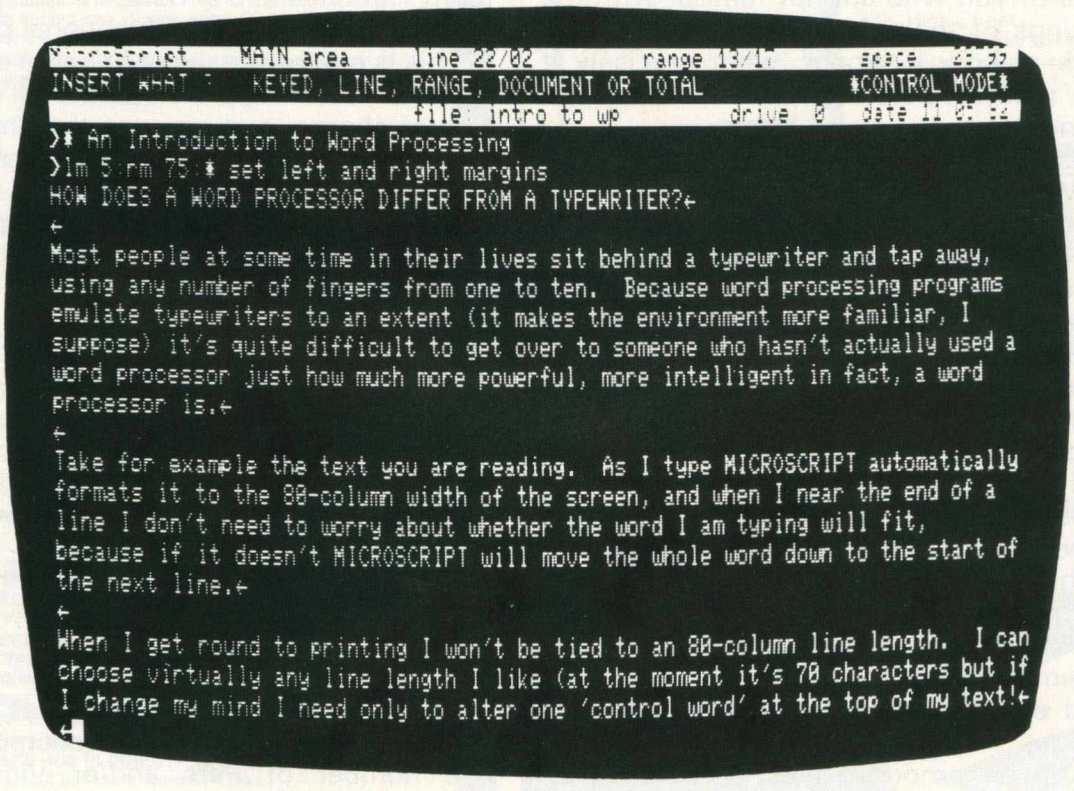
So, an active group based down in Slough, who are doing a lot of good work on behalf of Commodore, and who are providing a valuable service for both newcomers and old hands at the personal computing game.

If you'd like your club featured in a future issue of Commodore computing, please get in touch with the editor of the magazine.

MICROSCRIPT

Word processing made simple!

YOU DON'T HAVE TO BE A WIZARD TO USE MICROSCRIPT



Whether you're a secretary, businessman, author, or journalist – MicroScript can help you. From a single page letter to volume mailings, from one page reviews to complete books – MicroScript is simple to use yet rich in facilities.

MicroScript has the largest text memory capacity of any word processor for the Commodore 8000 series computers – even though it's probably the longest program ever written for the 8032 and 8096. Sounds incredible doesn't it?

Here's how we do it: MicroScript is the first in a new generation of business software, supplied not on a floppy disk but burned into banks of micro-chips. What this means is that instead of using the memory that's already there, MicroScript adds 46k of ROM and 2k of RAM to your computer. It even doubles the ROM expansion capability.

What does all this mean for you, the user? It means that you're investing in the latest technology. It means that you're buying the Rolls-Royce of word processors. Most of all it means that you're investing a little money to save a lot of time. Simple, isn't it!

MICROSCRIPT IS AVAILABLE FROM YOUR COMMODORE DEALER

© Viza Software 1982

Distributed by SUPERSOFT, Winchester House, Canning Road, Harrow, England
Telephone: 01-861 1166 (3 lines)

Education

Educational Workshops

Last month we gave a rundown on just some of the suppliers of educational software and hardware add-ons for both the Pet and Vic range of computers. However, there is another way of receiving software material, and moreover a way that is free of charge. This can be done by joining Commodore's Educational Workshop scheme.

For those of you who are not familiar with the inner workings of the scheme, this month we'll take a look at how it works, what precisely it does, and most importantly how you can go about joining it.

What is a Workshop

Basically an educational workshop, as defined by Commodore, is any educational establishment, be it a school, college, university or whatever, that either owns or has access to any number of Pets and/or Vics, and is prepared to assist others in the school or outside of it. Quite what form that assistance takes is up to the individual place concerned, but most commonly places will have open nights, or more formal training sessions, whereby newcomers to the microcomputing arena can see the kind of work that's being done, and how they could benefit as well.

So, if you belong to an educational establishment, and are prepared to assist others in this kind of manner, you're part of the way there.

However, there's more to being an educational workshop than just helping out other people.

Public Domain Software

Commodore have put an awful lot of software into the public domain i.e. people can make a copy of the disk (or tape) without fear of infringing manufacturers copyright, and then pass that copy onto others who might benefit from its use.

Where does this software come from? Much of it of late has been arriving from schools and colleges in Canada, and making its way via various devious routes into this country and the hands of Nick Green at Commodore. Quite why so much should come from there is a mystery, but nonetheless it arrives here.

Needless to say there are contributions from other sources as well, and the U.K. is by no means least in the list of suppliers of such material. Many of the programs will have already been used in an educational environment, although probably an equally large number will simply be demonstration or half finished programs that never saw the light of day.

Perfect Software?

Now Commodore themselves are the first to admit that this software is far from perfect. This is not the point: the point is that this software exists, and whether you use it as is, or more usually modify it for your own uses, it will save many an invaluable hour in the planning and development stage of educational programming.

This is one of the major benefits of being in the scheme. Commodore distribute all this software free of charge, and you can then make as many copies as you like for your own internal use. Thus you will be receiving literally hundreds of programs at no cost to you: an important factor where traditionally cost-conscious schools are concerned.

What the Scheme Does

Essentially the above: to assist in helping others, and to promote and distribute public domain software. There have been in the past, and no doubt will be again in the future, a number of special offers to establishments in the scheme. Discount on software buying, or hardware buying, are just two examples that spring to mind.

How do I join

There are no hard and fast rules about how a college or school goes about becoming a member. The number of Pets and/or Vics owned by existing members can and does range from one to over fifty, so number of machines is clearly not a premium. Similarly, not everyone can, or has the facilities to, offer help and advice to outsiders.

Consequently the final decision must rest with Commodore themselves. Clearly if you have a few computers, and can give assistance to others, it will be of help in the final evaluation, but none of this is of necessity a pre-requisite.

Although the man in charge of Commodore in education is Nick Green, your initial contact at the company should be with Jean Frost, at the traditional address of 675 Ajax Avenue, Trading Estate, Slough, Berkshire. Write and tell her that you're interested in the scheme, and require further details, and all the necessary information will be sent off to you.

If you join, it will cost you nothing, and I'm sure you will not regret the decision.

More on the world of computers and education next month.

from small systems.....

PET HARD DISKS

The Small Systems HARDBOX acts as an intelligent controller for up to Corvus Winchester drives.

- PET DOS 1 and 2 compatibility
- Multi user capability on Corvus Drives
- 16 Megabyte max file size
- 65535 max records per relative file
- Over 2000 files on 5 MB drive

Price..... £495

PET MINI-WINI

The MW-1000 Mini-Winchester is a compact desk-top unit that just plugs into the PET – gives you up to 12 Mb of hard disk storage under CP/M or PET DOS or both!!

At the flick of a switch, this amazing unit allows you to have:-

- The whole disk under CP/M (plus 60K of RAM and Z80)
- The whole disk under PET DOS
- Half of the disk under PET DOS and half under CP/M!!

No changes to the PET or ROMS are required – just plug in and go.

3Mb..... £2538
6Mb..... £2837
12Mb..... £3360

Prices include CP/M Utilities, 60K RAM and Z80.

PET CP/M

SOFTBOX and CP/M SOFTWARE

SOFTBOX allows the PET to run the worlds most popular operating system for micros. Operates with PET floppies and/or a Hard disk system.

Comprehensive range of CP/M software available – ask for our Catalogue.

Softbox (with RS232 + Hard disk interfaces as standard) ..... £495 *new low price!*

IEEE-488

IEEE-488 SERIAL INTERFACE TYPE C..... £120

IEEE-488 SERIAL INTERFACE TYPE B300..... £186
40 char input buffer

IEEE-488 ADDRESSABLE PARALLEL INTERFACE TYPE A100..... £106

TYPE G.P.I. AP MICROPROCESSOR BASED IEEE-488 BI DIRECTIONAL INTERFACE..... £275
IK input buffer standard 59K max.

TV/VIDEO MONITOR INTERFACES..... £46

RS 232C TO 20mA CURRENT LOOP ADAPTER..... £17.50

PETSPEED

Optimising Basic Compiler..... £240

DEVELOPMENT TOOLS

S10 /CP/M

8048 family in circuit emulator..... £550
8748 programmer..... £395
Prom emulator..... £295
Range of cross assemblers for most popular micros..... £95
8048 BASIC compiler..... £195

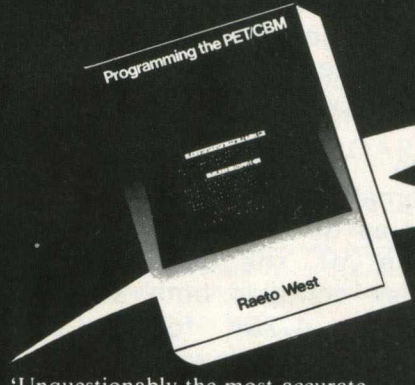
COMPLETE PET and HP-85 SYSTEMS

small systems engineering limited

2-4 Canfield Place, London NW6 3BT. Telephone: 328 7145 Telex 264538

NEW

Programming the **PET/CBM**



OFFICIALLY APPROVED BY COMMODORE

'This book is excellent.' - *Jim Strasma*

'Unquestionably the most accurate and comprehensive reference I have seen to date.' - *Jim Butterfield*

Bestseller — comprehensive teaching and reference book on all software aspects of Commodores 2000, 3000, 4000 and 8000 microcomputers and peripherals.

Many programs, charts and diagrams.
17 chapters, appendices, and index.
iv + 504 pages, 19 × 26 × 2½cm.
Paperback, ISBN 0 9507650 0 7.
Price in UK and Europe £14.90 each (incl. post and heavy-duty packing).
Five or more books at £12.90 each.
Soft plastic covers 45p each.
Dealer enquiries invited.
LEVEL LTD., PO Box 438, Hampstead, London NW3 1BH. Tel: 01-794-9848.

Cut out or copy coupon, or write to:
LEVEL LTD. (CC), PO Box 438, Hampstead, London NW3 1BH.
Send ..... copy/ies of *Programming the PET/CBM* at £14.90 (post free)

I enclose cheque/P.O. for £.....or official order.
NAME .....
ADDRESS .....
.....

Fast Service — same day despatch CC582

COMAL Part Three

Over the last couple of months we've presented two articles by Brian Grainger, on the use and implementation of Comal for the Commodore range of micros. No-one has yet done a Vic version, and to be honest I can't see any point in anyone producing such a version. Twenty two columns are not ideal for showing a structured language at its best. Still, you never know. This month, John Collins of Commodore's Technical department, fills us in on a few more facts and figures.

First of all, some basic (sorry!) information about the language Comal 80. Last Spring Commodore placed an implementation for its hardware (4032 and 8032) in the public domain throughout the world. The language was originally developed, principally by Borge Christensen, at the Danish State Teacher Training College in the mid-seventies. The name comes from COMmon ALgorithmic language.

The intention was to combine the simplicity of Basic with the algorithmic structure of Pascal: initially Basic was kept as a subset of Comal. In 1980 a working group defined an improved version, Comal 80, which no longer maintains Basic as a subset. It is now the first language taught to all Danish schoolchildren, and is the ideal first language for use in secondary education and at the top of primary education.

Commodore Version

Commodore's Comal 80 v.00.11 includes the full Comal nucleus as defined by the working group, plus some of the recommended extensions. The disk contains both a unified version, allowing only 4.75K for a Comal program, and a split version, allowing 15K. It is this version which has been distributed (on a 4040 disk and with a single version of a brief manual) to all of the Commodore educational workshops that had been appointed at the time.

Anyone can take a disk along to a meeting of their local workshop (details from Jean Frost on Slough 74111) and copy the Comal disk. Arrangements for obtaining a copy of the manual vary from workshop to workshop, but do remember that photocopying is seldom free. As there is one file on the disk which is 106 blocks long, the utility program supplies with your disk unit 'Copy 2040-8050' cannot be used to transfer Comal onto an 8050 disk: Jim Butterfield's 'Copy/All' (available through most regional ICPUG groups) has to be used instead.

New Enhancements

Brian Grainger, author of our last two articles, has modified Comal to work with cassettes, and also to work on the 3032. This version is available through the Independent Commodore Products Users Group. ICPUG members (subscription 7.50 pounds per annum to Jack Cohen, ICPUG Membership secretary, 30 Branchester Road, Newbury Park, Ilford, Essex, IG2 7EP) can obtain it by sending a blank disk or cassette (as required) together with payment for return postage to Bob Wood, ICPUG Software Librarian, 13 Bowland Crescent, Ward Green, Barnsley, South Yorkshire, S70 5JP.

JKL software in Denmark have enhanced Commodore's Comal 80 to rev. 1.01 which includes all the recommended extensions, and has been described by Borge Christensen as far and away the best implementation of Comal on any machine. A soft-loaded version for the 8096 is in the public domain worldwide: it comes on an 8050 disk, and allows 38K for a Comal program.

Copies can be obtained by sending a blank disk with payment for return postage to the Comal User Group, c/o North London Hobby Computer Club, Polytechnic of North London, Holloway Road, London N7 8DB.

To use this extended Comal in any model other than the 8096 an add-in hardware board is required. This works with any Pet from the 8K 2001 (but with new ROMs) up to the 8032, enabling the Pet to power up into Comal (but with Basic still available). Further details are available from Instrutek A/S, Christiansholsgade, DK-8700 Horsens, Denmark (tel. 010 45 5 61 11 00).

End user price of this board is 495 dollars: contact Instrutek for U.K. pricing, and a 1-off dealer discount of 15%. They also have available an extension board providing high resolution graphics (512 by 256 pixels) from within Comal. The price of this is a further 495 dollars.

Documentation

Commodore's original Comal manual is available by taking a copy via a software workshop. A new version including the extensions now exists, but it is only being issued to workshops in response to specific requests since Len Lindsay, founder of the USA Comal User Group, has written what will surely be the standard Comal manual, published by Reston (a subsidiary of Prentice-Hall International) and distributed via U.K. bookshops.

As well as this there are a number of books on

Comal which are now making their appearance.

The first of these is Structured Programming with Comal, by Roy Atherton, which we took a look at last month: it costs 6.90 pounds in paperback, from Ellis Horwood Ltd. Borge Christensen's own Comal Tutorial book (10 pounds in hardback) is now also available from Ellis Horwood Ltd.

Another little number which would be worth investigating by anyone remotely interested in Comal is the Comal Bulletin. It boasts an impressive editorial line-up, starting with Roy Atherton as senior editor, and amongst the consultant editors are Borge Christensen, John Collins (had to give you a plug John!), Len Lindsay, et al. Articles have and will include The World of Comal, The History of Programming Languages, Keeping Control of Long Programs, Comal Standards, Comal Structures, and many more.

Subscription to this bi-monthly publication is 10 pounds in the U.K., and 13.50 pounds overseas, and it is again available from Ellis-Horwood Ltd., at Market Cross House, Cooper Street, Chichester, West Sussex, PO19 1EB.

User Groups

As well as Len Lindsay's group in the States, details of which appear in his quarter page advertisement in every issue of Commodore Computing, there is now a UK Comal User Group. They meet from 7 to 9 pm on the first and third Wednesday of each month (but during Polytechnic term time only) in Islington Community Computer Centre at the Polytechnic of North London.

Anyone with queries on Comal is quite welcome to contact the Polytechnic, by writing to:—

Barry Miles,
Comal User Group,
c/o North Longon Hobby Computer Club
Polytechnic of North London
Holloway Road
London N7 8DB

We'll be back next month with more news on views. If there's any particular aspect of Comal you'd like to see attacked here, please get in touch with the editor, and we'll see what can be done.

NEW

COMMODORE PET COLOUR GRAPHICS PROCESSOR

- 640 (H) 288 (Y) × 3 planes
- 8 colours, double-buffered screen
- 16 bits microprocessor (intel 8088)
- 192 kbytes of dual-ported display memory
- standard graphic functions
- program in PET Basic

14in. COLOUR MONITOR

- high resolution
- RPG inputs

AVAILABLE NOW

Colour Graphics Processor: £599+VAT
High Resolution 14in Colour Monitor: £620+VAT

Enquiries and Demonstration invited
CAPITAL COMPUTER SYSTEMS
32 Windmill Street (Off Tottenham Ct. Road)
London W1P 1HH
01-636 3863 (3 lines)

PINEWOOD COMPUTERS
announce
the launch of
the 64K EXPANSION BOARD for 8032 PETs

Yes. We couldn't wait for the others so we have launched our own 64K Memory Expansion board to upgrade the 8032 PET to a full 96K. Silicon Office and other 96K programs are now possible on a 32K PET with our board. It is of U.K. design and manufacture and comes complete with full fitting instructions.

Our price £350

Other new PET enhancements include:
EPSON/PET INTERFACE CARD **RRP £90**
For all MX printers. Our board gives 40 column PETs uppercase and graphics and 80 column PETs both upper and lowercase without the need of switches or any software routine.
RICOH RP1600 INTERFACE CARD **RRP £115**
Our board gives 40 column PETs uppercase and 80 column PETs both upper and lowercase without any restrictions.

Add £10 delivery plus VAT to above prices.
To place your order send your remittance for the required amount to:

PINEWOOD COMPUTERS
Mail Order Dept.,
17 Adelphi Crescent,
Hayes Park, Hayes, Middx
or telephone 01-841 1507

DEALER ENQUIRIES WELCOME

Microcomputers

A Beginner's Guide

This month we start a brand new feature in the magazine, aimed exclusively at the businessman and the world of microcomputers.

Each month we'll be looking at the various aspects of computers in business, not only as a guide to the newcomer, but also intended for the more experienced user who is perhaps trying to increase the potential of his or her system.

We'll round off each article with a brief look at any new developments in this field in the software, hardware and also book areas.

Where do I begin?

This must be the most commonly asked question by the businessman who wishes to increase the efficiency of his office. (At the extreme risk of offending all womens libbers, I'll continue to use the term 'his' from now on. Please take it to mean both sexes, I can't keep writing 'his or her' all the time!). There is an enormous number of computers on the market, each one seemingly backed up by a dazzling array of software and various users guides. Truly, where do you begin?

We are not a general magazine, we are solely covering the Commodore range of machinery, so our choice is made significantly easier from the outset. We will only categorise various job needs into one or more of the Commodore computers. After all, for every micro currently available, there will be, somewhere along the line, a Commodore machine that will be reasonably analogous. Consequently, we will consider everything from that company's viewpoint.

The first thing that must be decided is, what do you want the computer to do? That is as hard for you to decide as it is for the salesman to answer the question ''Well, I know it's a computer, but what precisely do they do?'' The oft-quoted answer ''Anything you want them to, sir'' is insufficient for our needs, just as perhaps your idea of ''I want it to do my accounts'' is for the salesman. They say forewarned is forearmed: this is what you must do.

Define Your Terms

As in any argument, you must first of all define your terms. Take a look around your office, and try and assess precisely what you want doing. Perhaps you perform a lot of mailouts that you personalise because you think it will impress your clients. It certainly will, but it will not impress the secretary if it means typing out virtually the same letter one hundred times. Thus you will need some kind of word processor, but this is not enough.

There are many word processing packages available at present, ranging from the extremely basic to the very sophisticated. For the purpose of a plain mail-out, a fairly simple word processor would suffice, but you may want to do more. A common event is to install a package and then discover that you want it to do an awful lot on top of what you actually bought it to do. It is no use then saying ''Oh if only I'd spent another 50 pounds and bought a decent package''. Thus, you ought to consider what other common office procedures could be handled by a word processor, and go armed with that information as well.

Obviously, word processing is not the only kind of office or business procedure that could be computerised. Other common examples include the inevitable accountancy, stock control, payroll, and so on, but don't forget that there are many other problems that you may be encountering that could easily be solved by the introduction of an appropriate data base system.

For example, for a long time people used to bemoan the fact that there was not set program for estate agents, forgetting that a good data base could quite easily cope with questions such as 'give me all the three bedroom houses in Croydon currently selling for under 33,000 pounds that have a garden gnome next to the goldfish pond', or whatever. Many office practises can be tackled in this way.

As we stated earlier, make sure you know what you want before entering the showroom.

Close Encounters of the Salesman Kind

When you enter the showroom, be sure to be armed with a series of questions that you will not be sidetracked from. In other words, do not go in there with the intention of purchasing a payroll for 100 employees, and come out with a Pacman machine to keep 100 employees deliriously happy.

There is no doubt about it, a salesman is paid to be a salesman, and if you're hovering between purchasing or not, he will be perfectly capable of pushing you into a buying situation. ''Ah yes sir, if you purchase this program AS WELL, you'll be able to . . .''. This is not the reason you walked into the shop: you want to tackle specific problem areas, and not wander off into some great debate about the merits of any particular company's product.

Make no mistake, I am not decrying salesmen. Rather, keep your wits about you: it's your money after all, it's only his commission.

Get to the Point

You know what you want, and it's up to you to convince the other person THAT you know what you want. "I want a program (or programs) that perform the following functions", and make sure you see the programs in action before going any further.

It is a common practise, and regettably quite a deplorable one, that many packages have their own demonstration programs ready written, and when performing to that demonstration can appear to be perfectly adequate for your purpose. Do not ask to see (or be pushed into seeing) a demonstration using someone else's data: if you want a computer to look after your stock control, for instance, go armed with data of your own.

After all, it's your data the program will be handling, not someone else's. Certainly, give realistic figures, at the same time looking ahead to what your future requirements might be, but the message has to be: it's your office, it's your money, so get the system you want and not the one the salesman wants you to have.

Summary

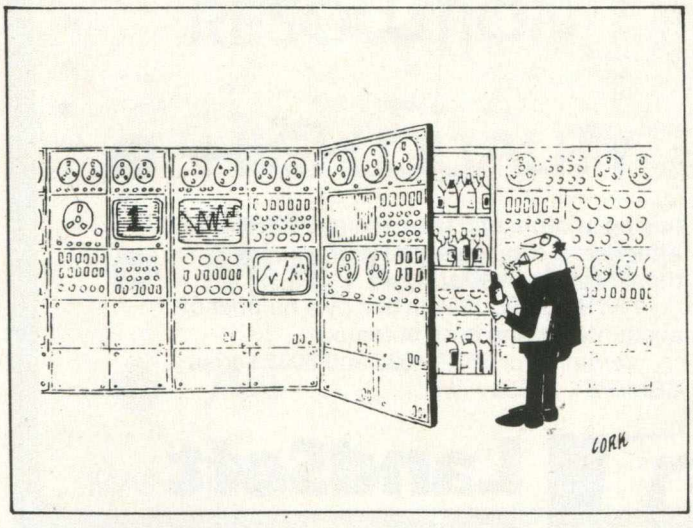
When purchasing a computer system to perform whatever function, remember that you are the one who is going to have to use it. If not personally, you will certainly be held responsible for any shortcomings that might expose themselves as time goes by. Don't worry that you're taking up too much of someone else's time: that's what they're paid for.

Next Month

These pages have only featured a general guide on purchasing your first system. Over the months ahead we'll be exploring in greater detail more specific application areas, and covering the usual (and perhaps not so usual) office practises that can quite easily be computerised, saving you not only time, but more importantly money.

New Hardware

You'll no doubt have read about the plethora of new machinery that Commodore have committed for release in the coming months. The news on this is very promising: a recent visit to



'Oh, I Don't Mind the $800,000 Price. But is that the Only Colour it Comes In?'

Microcomputers

Commodore left this author convinced that for once they have got it right, and that the announced new products will actually appear on time.

The main machine as far as we are concerned is of course the Commodore 720, and to a less extent the 510 series. Clearly both of these are aimed at the top end of the market, with main emphasis on the 720. This is determined to win back the position that they were possibly losing to Sirius (and others), and stands an extremely good chance of doing so.

The software planned for this is aimed fairly and squarely at the business end of things. As we mention elsewhere, some five companies have 720s at the moment: one of them, MMS in Bedford, are developing an integrated accountancy package to run on the beast, and if it lives up to the reputatin of the Microfacts82 system, will certainly prove a welcome member of what is bound to be a fairly large software family.

The 8000 machines of course keep rolling on, with plenty of existing products to keep them going, which brings us nicely to . . .

New Software

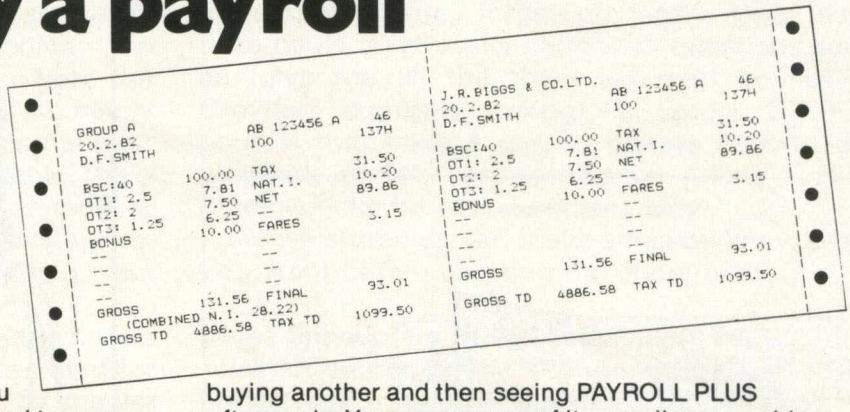
Not too much to report here, as not too many new business software packages have appeared lately: rather it's the old favourites staying in the game. However, keep an eye out for Swedish firm Datatronic, who are bringing out increasing numbers of packages, with the emphasis going more and more towards the business side.

New Books

The best buy at present appears to be a book called 'Business System Buyer's Guide', published by Osborne/McGraw-Hill at a price of 5.95 pounds, and written by Adam Osborne with Steven Cook.

Although this book is aimed at microcomputers and first time buyers in general, it does give a wealth of useful information about pitfalls to avoid, and practises to use when deliberating over which system to buy. So, although we know the computer make we want, we do not know which particular model or package to go with it, so this is certainly a book worth exploring. It contains much useful information.

How to buy a payroll program...



First, go to your CBM/PET dealer and ask to see at least two different payrolls. Second, make sure that one of those you see is the LANDSOFT 'PAYROLL PLUS'.

We are serious when we say you should see more than one. That way you are more likely to find the one most suited to your needs. However good PAYROLL PLUS is, there are certain things it won't do that other payrolls will — and vice versa.

Why do we say that you should insist that PAYROLL PLUS is among those you should see? Because it is so elegant in operation and so extremely easy to use. If you want a payroll that needs an expert 'computer person' to operate it, or if you enjoy spending hours trying to decipher an operating manual then PAYROLL PLUS is definitely NOT for you. You would find it too quick and easy to master.

So don't make the mistake of buying another and then seeing PAYROLL PLUS afterwards. Your awareness of its excellence could then be most frustrating!

PAYROLL PLUS is in use by a number of accountants and even bureaux.

Versions for 8032, 4032 and 3032 series CBM/PET **£150 + VAT.**

LandSoft.

28 Sheen Lane,
London SW14 8LW.
Telephone: 01-399 2476/7

SUPERIOR PROGRAMS FOR THE CBM/PET MICROCOMPUTER

HP's NEW LOW COST COLOUR PLOTTER

- FAST PLOTTING
- HIGH RESOLUTION
- HIGH QUALITY
- MULTICOLOUR
- A4
- COMMODORE-COMPATIBLE
- APPLE-COMPATIBLE
- NATION-WIDE SERVICE
- UNDER £1060
- RS 232 AND HP1B INTERFACES

MAIN DISTRIBUTOR

DEALER, O.E.M AND CONTRACT ENQUIRIES WELCOME

THE HOLDENE GROUP

Bray House,
Leicester Place,
Leeds, LS2 9EH
Tel: 0532 459459.

48 Great King St,
Edinburgh,
EH3 6QY.
Tel: 031-557 4060.

82A Water Lane,
Wilmslow, Cheshire,
SK9 5BB.
Tel: 0625 529486.

Software Review

Microfacts82

Many microcomputers find their homes in the business environment, fulfilling a wide variety of tasks. One of the most common practises is in accountancy, where many man-hours can be saved by judicious use of hardware and software. This month we take a look at one of the major accountancy packages on the market at the moment, namely Microfact82, from MMS in Bedford.

Facts 'n' Figures

Microfact82 is a fully integrated sales, purchase, nominal and VAT ledger system, retailing at 1,000 pounds plus VAT, and will run on the 8032/8096 with an 8050 disk drive, and a variety of printers. It is in fact an update to their earlier Microfact80 package.

Although their sales handout states that all up-dates and maintenance are provided free, with no time limit, the dealer hand out states ''We shall supply the up-date free of charge, but we have no objection if you wish to charge the user''. If you are a user, you ought to have an objection if anyone attempts to charge you!

As a quick footnote, MMS also provide two other packages which link into this main one. These are Microfact82 Sales Invoicing, which links into the Sales Ledger part of the main program (and has the facility to link into the sales, nominal and VAT ledgers), and costs 250 pounds plus VAT.

The other program is the Microfact82 Stock System, which like the Sales Invoicing, is compiled, and links into everything else. It gives you the facility to prepare an invoice, update the sales ledger and stock records simultaneously, and finally interface to the nominal ledger. Quite impressive, at 500 pounds plus VAT.

Further Improvements

The original Microfacts was introduced in early 1981 for the 3000 series of machines, and when the 8000 appeared later that year, many changes were made to the existing programs to take advantage of the new facilities now offered them. Chief amongst these changes was the use of compiled programs, which speeded up execution considerably, and a complete revision of the menu structure, giving much faster access to the independent routines.

Capacity

The total number of records available using an 8050 disk system averages out at 10,000. However, this is made up of sales accounts, purchase accounts and nominal accounts, along with outstanding sales and purchase transactions and period nominal transactions. The former take 2.5 records per account and the latter 1 record per transaction. Thus any combination, depending on company activity, can be calculated from the above figures.

Security

There are a number of built in security checks, to avoid unlawful tampering with your company data. For instance, all nominal ledger reports can be protected by a 4 digit password number. In addition to this, there is an extra password number included, which allows access (provided you know what you're doing) to the password change routine.

Program Operation

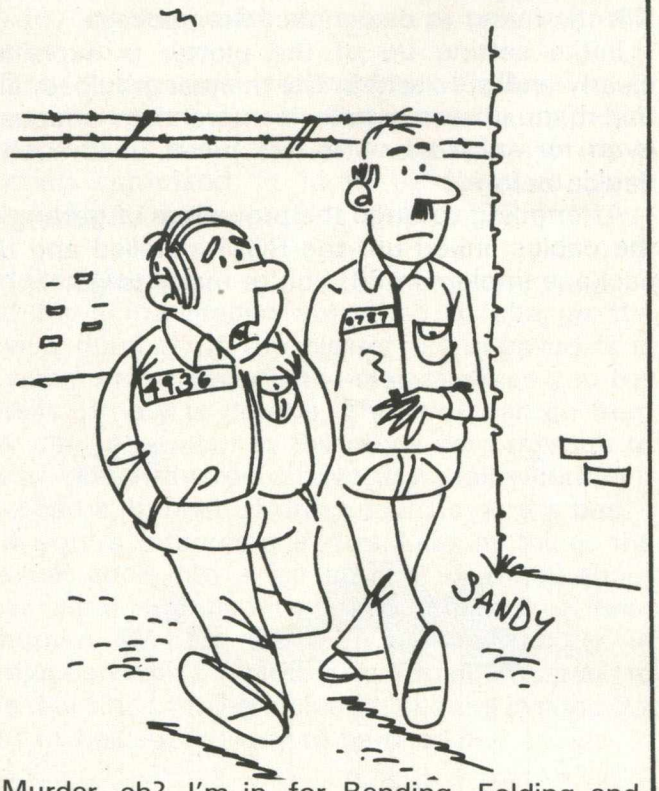
The programs are extremely easy to use, and unlike most accountancy systems do not require hundreds of disks to be swopped continually. Microfacts uses just two disks, but of course it is desirable to have backup copies made at fairly frequent intervals. There's no point in computerising your accountancy, only for someone to come along and accidentally wipe out your data.

To do this from within Microfacts would be rather difficult, if not impossible. However, there's always the risk of accidental erasure from exposure to magnetic fields or whatever.

The documentation as a whole is quite good, but as most of the instructions for use can be displayed on the screen, there is no need to keep constantly referring to 'War and Peace' as is the case with a number of other packages. The users guide is very good: it covers everything in a pretty straightforward manner. I am by no means a professional accountant, but it was all simple enough to understand.

Setting Up

Obviously a great deal of time will be spent initially entering all your previously man-recorded data. Logically enough, the user will have to exercise a fair amount of caution here, as one slip-up could eventually prove disasterous. However, there is a lot of checking of data inputted by Microfacts, field lengths are displayed, all necessary instructions are shown on the screen, and so on, so there should be no great problems encountered here. As a whole, the package is user friendly in this, and other, aspects.



'Murder, eh? I'm in for Bending, Folding and Mutilating Computer Cards.'

Ledger Control

Having set up the company account, of which up to five can be stored per disk depending on overall volume, ledger controls then have to be implemented. This is done by specifying company name (up to a maximum of five), ledger number (five sales or purchase ledgers per company), ledger name and type (whether sales or purchase), and finally nominal ledger control account numbers.

If the ledgers are to be linked, then the program will automatically create the relevant control accounts in the nominal ledger and provide the relevant account descriptions.

Having gone through this systems control procedure, the individual accounts are then set up. This is done quite simply by returning to the main menu and selecting the appropriate option, and one then follows normal accountancy practise, with no deviations to accommodate the computer.

Once you're set up and ready to go, starting up every day is a straightforward procedure. Various printer options are supported, which have to be specified at the commencement of any run. The paper to go with this can either be standard 11'' by 9.5'', or specially printed sales invoices, sales ledger statements and purchase ledger remittance advices.

Samples of these can be obtained from any Microfacts dealer. Alternatively you can always have your own personalised paper printed, but this does tend to be on the expensive side.

Various standards for data entry, both alpha and numeric, are adhered to throughout: again, once the user is familiar with these, no problems should be presented in the day-to-day entering of information.

Period End Routines

Having taken the obligatory security copies of the master disks, in case of disaster, the daybook file is then run out, which automatically updates the nominal and VAT aspect of any transactions. This can in fact be done at any time, but is preferably only done at the end of the period.

Cash book, petty cash book, wages etc. are entered as self balancing journals, followed by a printing of the trial balance until one gets it right. The period nominal ledger transaction report follows, without which the period end routines cannot be run.

Finally the period nominal journal report (and sales and purchase reports if necessary) follow, before the actual period end routine. This removes all fully settled transactions from the sales and purchase ledgers, and clears the period nominal ledger transaction file, whilst modifying the nominal ledger account balances brought forward.

Conclusion

It would be impossible here to go through thoroughly everything which is implemented in this package, otherwise there would be nothing else in the entire magazine! Suffice it to say that it covers everything desired of an integrated accountancy package, with links to stock control et al. as optional extras.

It is not cheap: no package of this nature and scope can ever hope to be so. For the price you pay, you are getting a very good deal.

To sum up, one of the best and most thorough accountancy packages on the market at present. Ring up Shane Barnes of MMS (0234-40601) to find out who your nearest appointed Microfact82 dealer is, and get them to take you through its paces. If accountancy problems are clogging up your company, or even if things are going fairly well, you will not be disappointed, and you may well find yourself saving both time and money. After all, isn't this what accountancy is all about?

Hardware Review

PD Digital Plotter

Last month we featured a review of the HP7470A plotter from Hewlett Packard. Coming in at over 1,000 pounds, it nonetheless proved to be an attractive buy for the kind of use it would get put to. Can a graphic plotter at less than 600 pounds prove equally attractive?

J.J. Lloyd, down in Southampton, have been in business for around 20 years now, and have been producing analogue recorders for the last 12 of those. As well as these, their main areas are in tensile testing equipment, and educational equipment.

Already well-established as one of the largest European supplier of analogue recorders, they have now come out with a low cost digital plotter (a spin-off from earlier work) for the Pet. Incidentally, this is their first entry into the printer/plotter market.

Places of Use

As with most plotters, the main use of the PD4 would be to provide graphical representation in a fairly clear way, that could not otherwise be obtained on the more traditional printers available. Thus, anywhere where the Pet is being used to accept and analyse data in fact. Such fields as maths, physics, engineering, R. and D., architecture, and so on.

First Appearance

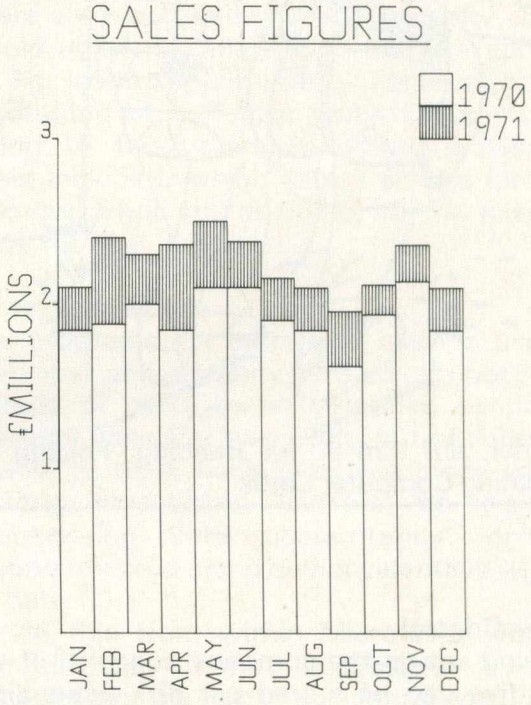
The machine is of attractive design, with a number of exterior controls for use in local mode: one of these toggles you from remote to local. Of the others, two are for use in defining the zero point for the X and Y axes, one is a pause facility whilst plotting (and unlike the HP one, this works properly! i.e. it does start off in the same place as it left off), one resets the axes after you've finished re-definition, and the final one is a simple lift and put down the pen.

Linked up to the Pet by a direct IEEE connection, the plotter will work with any of the 3000, 4000 or 8000 series machines. Interestingly enough, the software to drive the beast comes in ROM form, which adds a number of commands to the existing Basic interpreter, performs straight line interpolation and allows character generation.

The idea of putting the software in ROM is a good one, as it means that the Pet can still continue to operate as a normal machine, but access to the new commands is but a SYS call away. To get back into normal mode, there is a Kill command to disconnect the package.

Initial setting up of the plotter is explained clearly and precisely in the manual provided, and the manual as a whole is more than adequate even for someone who has never used such a device before.

After going through the procedure of getting all the cables linked up, the ROM installed and the package implemented, you're ready to go.



Commands Implemented

Fifteen extra commands are added to the Basic interpreter.

Whilst this is not comparable to the two million with the HP machine, it is enough to produce reasonable output, as the accompanying illustrations show. All the now-to-be-expected commands are there, such as move pen, draw (both of these in absolute or relative), print various strings, specify size of characters, draw axes, home, and so on.

Six additional error messages are also added to the command set.

A couple of special commands deserve extra mention. The first of these is rotate character, which can be done through 0, 90, 180 or 270 degrees. This is particularly useful for such functions as plotting graphs, labelling axes, and so on. Like all the other commands, it has to be reset before proceeding further, otherwise some very peculiar results can occur!

The other interesting one is for producting

special graphics characters: a set of four are already implemented as soon as you set the system up.

One idiosyncrasy with mixing with Basic must be mentioned here: when using an IF . . . THEN statement, if the THEN is to be followed by a plotting command, it must be followed by a colon. Thus, we have IF . . .THEN: !DR etc.

Detailed Useage

If for some reason you wish to change the device number of the plotter (as supplied it is device number seven), the manual gives you the details of how to do this. Thus you can go from any device number in the range zero through to thirty: obviously you'd have to be fairly careful if you had a number of other devices on the bus.

If you're interested in this kind of thing, the manual goes into a fair amount of detail about how data is transmitted and received, a brief rundown on how the IEEE standard works in conjuction with the plotter, and so on. It's good to see this kind of thing included, bearing in mind the sort of fields it is likely to be used in.

Sample Programs

Whether J.J. Instruments supply sample software on disk or tape is up to how you charm the salesman. We were certainly provided with such material, and I should imagine they would have no objections to supplying you. Certainly a number of programs are included at the back of the manual.

Vectors are drawn in either absolute or relative mode: absolute relates to the origin, wherever you've defined that to be, and relative refers to the last pen position. Commands can also be stacked, so as to save repeating the new Basic command all the time. When using these new commands, one must always use variables rather than absolute addresses, so these must be defined before attempting to do anything.

Plotting Statements

For instance, the statement X=10:Y=20:!DR,X,Y would be acceptable, but not !DR,10,20. This in reality is of benefit, as it makes error checking of programs that much easier.

All of the variables you define refer to millimetre measurements, using an overall scale plotting area of 250 millimetres in the Y axis, and 180 in the X. The actual resolution of the plotter is 0.044 millimetres in the X direction, and 0.061 in the Y.

The sample programs provided cover such topics as drawing curves, printing of various special characters, axes, circles, and onto such esoteric subjects as Lissajous figures, Polygons, and Histograms. Given the basic information gleaned from this, it is easy enough to get heavily involved in your own plotting routines.

Technical Information

Some of this we'll have given you already, but one or two other salient points. The plotter will accept copy up to a size of 297 × 210 millimetres, but will only plot on a 250 × 180 grid. Plotting speed is, naturally enough, dependent on the speed of the Pet, but with pen raised it races along in the Y axis direction at a rate of 600 millimetres per second (or 1.34 miles per hour, if you want to impress your friends).

The pens are the usual plotter fibre tipped ones, and by judicious use of the pause facility, multi-colour plots can easily be performed. Both pen and paper can be obtained without any major problems.

Physically, it weighs in at approximately 9 kilograms, with dimensions of 360 × 445 × 125 millimetres. All in all quite a compact little unit, and certainly transportable.

Conclusion

There are quite a few plotters putting in appearances for the Pet these days, with more and more coming on the market all the time. Consequently for a first time buyer in this field, the decision as to which to choose is becoming ever more and more difficult: rather like a first time computer buyer, there is an increasingly large array to choose from.

Certainly, price will (as ever) be a major consideration here J.J. Instruments cannot be faulted. At just under 600 pounds it represents an attractive buy.

Ease of use must again be a major selling point. Whilst the J.J. one has nothing like the range of commands that the HP plotter we featured last month has, those that are there are probably sufficient for most purposes, and are as easy to use as it is to write a Basic program to print out the numbers one to ten. The HP7470A is easy to use as well, but rather like the difference between Basic and machine code, I feel the PD4 is a more comfortable introduction for the beginner.

All in all, the plotter (given the price) represents good value for money, and is definitely worth comparison with plotter retailing at twice the price.

For further information contact Dave Sawyer of J.J. Lloyd Instruments on 048 95-4221.

Book Review

Pet Fun and Games

The book has something for every games fan, action games, puzzles, games of risk, strategy, chance and fun, so no one who buys this book loses out.

Compiled by Ron Jefferies and Glen Fisher from programs sent in to the American cassette magazine Cursor over the last three years. Thirty-one games (and fourteen of the best programmers!) have been chosen, and they are all highly enjoyable either for their fun value, skill, creativity or even education value.

The contents are listed out at the beginning of the book in sections for the different types of games, and each game has a few words on what it does so you know what you have in store if you play it!

Although distributed in England it was written in America and one or two problems that might arise if you have certain Commodore Machines should be stated.

Mixing Business with Pleasure

In America Commodore marketed the CBM Business machine for the business man and the PET for the general market. On the CBM there are no graphic signs on the keyboard although the graphics are still there.

The book tells you how to get into graphics mode clearly and precisely, by using the POKE 59468,14 (12) method, but unfortunately if you have an 8000 series machine or a CBM using this program and run it your graphics will appear disjointed, although this does not stop you playing the game it does spoil the interesting graphics.

Introduction

The introduction is short and to the point, no messing about here with long words or waffle. The importance of any graphic symbols that appear in the listings is described in the introduction, i.e. every symbol reproduced in a listing is meant to represent one of the existing Commodore graphic symbols. As not every make of Commodore machine has these symbols represented on the keyboard legend, and even if they have some of them take a lot of finding, this is a useful convention for publishers to adopt.

One problem I have often come across is that of trying to read badly typed or even worse badly written programs! I cannot fault this book where the typing of programs is concerned. The typeface they use is large and very clear, one could not make a mistake through poorly typed listings. The printing itself is also very clear and remains so throughout the book, thus making the tedious chore of typing in a listing almost enjoyable!

Conclusion

A very good and enjoyable book, I cannot fault it. It is easy to understand with no complicated wording for the beginner. It is well typed and printed and at £7.50 it is reasonably priced for the young games player (Fathers, guard your wallets!). Available in this country from Osborne/McGraw/Hill, on 0628-23431.

Hands-On Basic with a PET

Hands-on Basic with a PET is just the book for all those of you who know nothing about programming the PET computer but want to learn. Although the books starts you off with a simple and straightforward introduction to the PET, it is very comprehensive and by the end you will have a thorough knowledge of how to program, as well as all the other subjects the book covers.

The author Mr Herbert D Peckham starts off by telling us the origin of the language Basic: no real bearing on what you will actually learn, but nonetheless quite interesting to know, and a useful background reference point.

The table of contents is well set out with good headings and simple sub-headings: the index supplements this admirably by providing clear and precise reference to material contained in the book. A quick look at these contents gives you a pretty good idea of the amount of ground covered here: from REM to Random, and stopping off at many places en route.

Overall Layout

Each chapter is neatly laid out, with questions and listings NOT squashed together but spaced out for ease of reading. An important point to notice is that the listings have not been re-typed especially for the book, thus causing more chance of errors, but instead the author has taken already tried and tested listings and printed them straight in 'as is'. A wise move, as there is nothing more annoying, whether you are in a classroom situation or at home, than typing in a listing only

to find that somewhere you have typed it in wrongly, and even more aggravating when you discover that it is not your fault in the first place.

The book has been broken up with diagrams and humourous quotes for assistance in remembering the fundamentals, and very quickly changes the easy terms that it substitutes in the early stages for the correct computer terms that you will have to use when working with computers.

Structured Programming

When the author does begin to delve more deeply into programming he attempts to teach a structured programming method, rather than the thud and blunder style currently employed by so many other computer books on the market.

Hands-On Basic with a Pet uses many examples, quizzes and tests throughout its content, giving the various answers at the back if you feel like cheating. Therefore, the best way for the learner to use the book is in conjunction with the PET itself, rather than just wading through it and trying to learn programming that way. An old Chinese proverb seems very much the basis for Mr. Peckham's philosophy:—

I am told and I forget
I see and I learn
I do and I understand.

SUMMARY

A very good book, I would recommend it for computer courses within schools and colleges. It is clearly and concisely written for teaching students who know nothing about the PET Computer, but wish to learn without the teacher having to juggle between five or six different books.

At £11.95 the book is fairly priced both for schools and colleges and for the home user.

McGraw-Hill (UK) Ltd do sell direct to the public as well as through mail order, at the following address:—

Marketing Services Dept
McGraw-Hill Book Co (UK) Ltd
FREEPOST
Maidenhead
Berkshire SL6 2QL

Their books should also be available through any good book shop.

MIDLANDS
COMMODORE PET SERVICE CENTRE

Phone Anne on 021-772 8181 about our

1. WORKSHOP & FIELD REPAIRS
2. BUSINESS SOFTWARE
3. STATIONERY & SUPPLIES

CBS CONSULTANTS
COMPUTER BUSINESS SYSTEMS

75 Watery Lane, Birmingham B9 4HW.
Telephone: 021-772 8181 (7 Lines)

Your move

Commodore CBM systems are dependable, efficient, versatile, easy to maintain. That's why so many companies, big and small make their move through

ALPHA Business Systems

Electron House
Industrial Estate
Church Street
Ware SG12 9ES

Telephone Ware (0920) 68926/7

Guest Expert

Dave Scott looks at Cursor Positioning Routines

After having programmed in a BASIC with many more extensions, structured constructs, and easy file handling commands, you might easily become frustrated writing code in Commodore BASIC. Of course, when working on the larger machines that support those types of BASICs, you don't ever get a feeling or understanding of how the operating system really works. One of the nice features about the PET is that you can actually look at the operating system, change certain parts of it to fit your own needs, and use some of the ROM routines to interface with your own programming.

One of the features I missed most was the ability to move the cursor to an X,Y location on the screen and print a message there. After a few attempts to position the cursor using POKEs to address 198 (position of cursor on above line) and 219 (line where cursor lives), I decided to write my own routine.

The routine is written in machine language and sits at the top of RAM, so that it can be used by any BASIC program that is loaded. The routine is entered via a SYS 32256, and your BASIC program must pass certain parameters as in the format:

sys 32256,y,x,''your message should appear here''

Where ''y'' is any line number between 0 and 24 (25 lines), and ''x'' is any column position between 0 and 79 (80 columns). If the format is not as that listed above, or your X,Y coordinates are not in the correct range, the routine will break with a ''syntax error''.

I have provided both a BASIC poke program that readjusts the top of memory in order to secure the routine and pokes in the appropriate code for the routine, and the actual source assembly listing for those of you who have an interest or wish to make changes.

Since it may be awkward to run the poker program everytime, you may wish to run it once, then break into the monitor and do a machine language ''save'' to save it in program format (for subsequent ''dloads''). In this case, I have also provided a loader program which first readjusts the top of memory pointers and the start of variable pointers (according to your largest program module in your application), then loads in the ''program'' version of the routine, and finally loads in your first application program, in this case, the ''master menu''.

This program will also work on 40 column PETs if you change the value ''80'' on line 200 of the BASIC poker program to ''40''.

```
00001  0000            ;*************************************************************
00002  0000            ;****                                                     ****
00003  0000            ;**** program name = display.src                         ****
00004  0000            ;**** author = dave scott, cbm  us                       ****
00005  0000            ;**** date = 02/23/82                                    ****
00006  0000            ;****                                                     ****
00007  0000            ;**** purpose: the purpose of this program is to print a message at ****
00008  0000            ;****          a specified x,y location on the screen.  it is       ****
00009  0000            ;****          entered via the "sys" command and the basic program  ****
00010  0000            ;****          must pass certain parameters.  the syntax for the sys ****
00011  0000            ;****          command is:                                ****
00012  0000            ;****                                                     ****
00013  0000            ;****          sys 32256,15,30,"this is test"             ****
00014  0000            ;****                                                     ****
00015  0000            ;****          where 32256 is the sys address, 15 is the line, 30 is ****
00016  0000            ;****          the column, and the string in between quotes is the  ****
00017  0000            ;****          message to be printed at that x,y location.  seal the ****
00018  0000            ;****          routine off from basic by doing a poke 52,126.       ****
00019  0000            ;****                                                     ****
00020  0000            ;*************************************************************
00021  0000            ;
00022  0000            ;*****************************************
00023  0000            ;****          kernal routines          ****
00024  0000            ;*****************************************
00025  0000            ;
00026  0000            outchr=$ffd2          ;print char in acc
00027  0000            ;
00028  0000            ;*****************************************
00029  0000            ;****      rom and system routines      ****
00030  0000            ;*****************************************
00031  0000            ;
00032  0000            chrget=$0070          ;char get (in 4.0)
```

```
00033  0000                chrgot=$0076           ;char got (in 4.0)
00034  0000                synerr=$bf00           ;syntax err (in 4.0)
00035  0000                ;
00036  0000                ;****************************************
00037  0000                ;****         storage areas              ****
00038  0000                ;****************************************
00039  0000                ;
00040  0000                txtptr=$0077           ;current char
00041  0000                chkspc=$007f           ;check for space
00042  0000                ;
00043  0000                ;****************************************
00044  0000                ;****         main routine               ****
00045  0000                ;****************************************
00046  0000                ;
00047  0000                      *=$7e00
00048  7e00                ;
00049  7e00  20 13 7e            jsr savcod                    ;save chrget code
00050  7e03  20 2f 7e            jsr movey                     ;cursor to line y
00051  7e06  20 39 7e            jsr movex                     ;cursor to column x
00052  7e09  20 43 7e            jsr prtmsg                    ;display message
00053  7e0c  20 24 7e            jsr getcod                    ;restore chrget
00054  7e0f  20 70 00            jsr chrget                    ;get next char
00055  7e12  60                  rts                           ;return to basic
00056  7e13                ;
00057  7e13                ;
00058  7e13                ;****************************************
00059  7e13                ;****     save chrget code 7f-80         ****
00060  7e13                ;****************************************
00061  7e13                ;
00062  7e13  a5 7f         savcod lda chkspc                    ;these lines save
00063  7e15  8d f6 7e            sta store                     ;the code from chrget
00064  7e18  a5 80                lda chkspc+1                  ;that check for a
00065  7e1a  8d f7 7e            sta store+1                   ;space and replaces
00066  7e1d  a9 ea                lda #$ea                      ;them with a nop
00067  7e1f  85 7f                sta chkspc                    ;instruction "ea"
00068  7e21  85 80                sta chkspc+1                  ;
```


DO YOU PLOT GRAPHS?

This is the affordable answer. The new *Hewlett Packard 7470A* two pen graph plotter connects directly to *your* Commodore Pet (no interface required).

For full graphic output of Graphs, Pie Charts, Histograms, Overhead Transparencies, also with full Digitising facilities.

Supplied complete with cable, programming manual and sample program listing at:

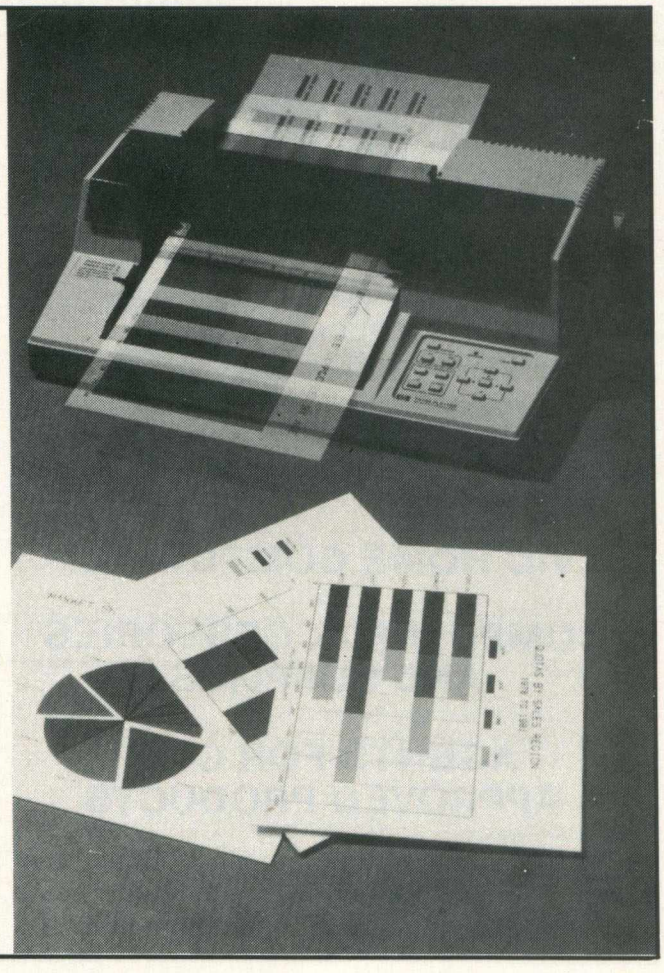
£999 + V.A.T.

10 days approval against official orders.

For further information or a demonstration contact:

C.S.E. (Computers)
12 Wokingham Road
Reading, Berks.
Tel: Reading (0734) 61492

P.S. Also available for Apple.



Guest Expert

```
00069  7e23  60            rts                                    ;
00070  7e24              ;
00071  7e24              ;*****************************************
00072  7e24              ;****  put back chrget code 7f-80      ****
00073  7e24              ;*****************************************
00074  7e24              ;
00075  7e24  ad f6 7e    getcod lda store                         ;these lines put
00076  7e27  85 7f              sta chkspc                         ;back the original
00077  7e29  ad f7 7e           lda store+1                        ;chrget code
00078  7e2c  85 80              sta chkspc+1                       ;
00079  7e2e  60                 rts                                ;
00080  7e2f              ;
00081  7e2f              ;*****************************************
00082  7e2f              ;****     cursor to line y              ****
00083  7e2f              ;*****************************************
00084  7e2f              ;
00085  7e2f  20 b8 7e    movey  jsr chkcom                        ;check for comma
00086  7e32  20 69 7e           jsr gety                           ;get value of y
00087  7e35  20 7b 7e           jsr posy                           ;cursor to y
00088  7e38  60                 rts
00089  7e39              ;
00090  7e39              ;*****************************************
00091  7e39              ;****    cursor to column x             ****
00092  7e39              ;*****************************************
00093  7e39              ;
00094  7e39  20 b8 7e    movex  jsr chkcom                        ;check for comma
00095  7e3c  20 90 7e           jsr getx                           ;get value of x
00096  7e3f  20 a2 7e           jsr posx                           ;cursor to x
00097  7e42  60                 rts
00098  7e43              ;
00099  7e43              ;*****************************************
00100  7e43              ;****        print msg                  ****
00101  7e43              ;*****************************************
00102  7e43              ;
```


KINGSLEY COMPUTERS LTD.
132 Desborough Road
HIGH WYCOMBE, BUCKS HP11 2PU

CBM BUSINESS SYSTEMS

VIC HOME COMPUTERS

COMPUTER ACCESSORIES AND SUPPLIES

AGENTS FOR CBM APPROVED PRODUCTS

COMMODORE SERVICE CENTRE

```
00103  7e43  20 76 00    prtmsg jsr chrgot          ;get current char
00104  7e46  c9 2c              cmp #44             ;is it = ","
00105  7e48  f0 03              beq prt1            ;  yes, print msg
00106  7e4a  4c b2 7e           jmp err             ;  no print syntax
00107  7e4d                    ;                           error
00108  7e4d  20 70 00    prt1   jsr chrget          ;get next char
00109  7e50  c9 22              cmp #34             ;check for quotes
00110  7e52  f0 03              beq prt2            ;  yes, go print
00111  7e54  4c b2 7e           jmp err             ;  no print syntax
00112  7e57                    ;                           error
00113  7e57  20 70 00    prt2   jsr chrget          ;get next char
00114  7e5a  c9 22              cmp #34             ;check for quotes
00115  7e5c  f0 0a              beq prtend          ;  yes, done
00116  7e5e  c9 00              cmp #0              ;  end of line??
00117  7e60  f0 50              beq err             ;   yes, print
00118  7e62                    ;                           syntax error
00119  7e62  20 d2 ff           jsr outchr          ;
00120  7e65  4c 57 7e           jmp prt2            ;get next char
00121  7e68                    ;
00122  7e68  60          prtend rts                 ;done
00123  7e69                    ;
00124  7e69                    ;****************************************
00125  7e69                    ;****    get value of y                  ****
00126  7e69                    ;****************************************
00127  7e69                    ;
00128  7e69  f0 47       gety   beq err             ;invalid char
00129  7e6b  b0 45              bcs err             ; print syntax error
00130  7e6d  20 c3 7e           jsr aschex          ;conver # to hex
00131  7e70  ad f5 7e           lda temp            ;temp=converted val
00132  7e73  c9 19              cmp #25             ;is y < 25
00133  7e75  b0 3b              bcs err             ;  no
00134  7e77  8d f3 7e           sta ycoord          ;store y coordinat
00135  7e7a  60                 rts                 ;
00136  7e7b                    ;
00137  7e7b                    ;****************************************
00138  7e7b                    ;****    move cursor to line y           ****
00139  7e7b                    ;****************************************
00140  7e7b                    ;
00141  7e7b  ae f3 7e    posy   ldx ycoord           ;cursor down using
00142  7e7e  a9 13              lda #19             ; ycoord as counter
00143  7e80  20 d2 ff           jsr outchr          ; home cursor
00144  7e83  e0 00              cpx #0              ;  is y = 0?
00145  7e85  f0 08              beq posyen          ;  yes, done
00146  7e87  a9 11              lda #17             ; 19 = home
00147  7e89  20 d2 ff    posy1  jsr outchr          ; 17 = cursor down
00148  7e8c  ca                 dex                 ;
00149  7e8d  d0 fa              bne posy1           ;
00150  7e8f  60          posyen rts                 ; done
00151  7e90                    ;
00152  7e90                    ;****************************************
00153  7e90                    ;****     get value of x                 ****
00154  7e90                    ;****************************************
00155  7e90                    ;
00156  7e90  f0 20       getx   beq err             ;invalid char
00157  7e92  b0 1e              bcs err             ; print syntax error
00158  7e94  20 c3 7e           jsr aschex          ;convert x to hex
00159  7e97  ad f5 7e           lda temp            ;temp=converted val
00160  7e9a  c9 50              cmp #80             ;is x > 80??
00161  7e9c  b0 14              bcs err             ;   yes
00162  7e9e  8d f4 7e           sta xcoord          ;store in xcoord
00163  7ea1  60                 rts                 ;
00164  7ea2                    ;
00165  7ea2                    ;****************************************
00166  7ea2                    ;****     move cursor to x               ****
00167  7ea2                    ;****************************************
00168  7ea2                    ;
00169  7ea2  ae f4 7e    posx   ldx xcoord          ;cursor to column x
00170  7ea5  e0 00              cpx #0              ;  if x=0, done
00171  7ea7  f0 08              beq posxen          ;  if not,
00172  7ea9  a9 1d              lda #29             ;  use xcoord as
00173  7eab  20 d2 ff    posx1  jsr outchr          ;  counter
00174  7eae  ca                 dex                 ;  29 = cursor ->
00175  7eaf  d0 fa              bne posx1           ;
00176  7eb1  60          posxen rts                 ;
```

Continued on page 53

Applications

"Specialist Data Logging"

Many industrial and research plants, and processes, must be monitored carefully, either for statistical information gathering or to warn of a malfunction in the system.

The programmable logic controllers normally used for these tasks are now being supplemented by the addition of a microcomputer. Woking computer and data logging specialists PPM Limited (Tel. 04867 80111) have developed a number of units designed for industrial data acquisition based on modified Commodore PETs.

PPM obtained one of the first PETs in the U.K. in 1977 for evaluation, and on the basis of their trials became Commodore agents, supplying around 1,000 PETs in the time since.

Their other interest in the PET lay in the fact that it used an IEEE interface for operation of its peripherals.

"It was the first low cost IEEE bus controlled micro-computer, and it revolutionised control monitoring, making data logging systems available to a much wider group of users.

"And as the cost of labour has risen, and the price of hardware dropped, the number of potential users has continued to rise.

"It makes sense to monitor processes automatically now, as it is by far the most cost effective method," PPM Director Roger Connel, explained.

The PET had the further advantage of being readily fitted with an RS232 interface, so PPM set about developing the PET into a control-oriented micro-computer.

This took two forms — a more or less standard looking PET, and a special 'industrial' rack system for use in factories.

Important modifications to the standard PET included the full implementation of the IEEE interface to suit control equipment; provision for the holding of programmes on EPROM as an option to floppy disc; and the designing of a programme which would load automatically when the unit was switched on.

The result is a system which to be as cost effective as possible takes standard PET peripherals — floppy disc drive and printers — and can be adapted to suit most industrial and research monitoring applications by varying the inputs, or by fitting appropriate test equipment.

The 'standard' model is intended for use in clean dust-free places while the rack system is for use on the factory floor where is could be subjected to considerable abuse in a harsh environment. In extreme conditions it can be fitted in a completely sealed cabinet.

It is in this sort of situation that the 'Autoload' facility is particularly useful. Starting the monitoring process is a simple matter of turning a key. This switches the assembly on and loads the programme on the PET, which continues to run unattended for as long as the power supply is on. If there is a break in power supply and the unit is turned off, it will automatically load as soon as power is restored.

The rack system is designed to be used with a plug-in keyboard, which is the engineering-oriented 4032 keyboard, even though the system uses the 8032 PET.

Use of the plu-in keyboard ensures that the programme is not subject to accidental change.

PPM will fit, as an option, a 20mA current loop to enable the PETs to be operated at a considerable distance from the data logging points, which may be scattered throughout a factory complex.

Sold under the name 'Compuscan', PPM data logging equipment has been installed in many different companies in the past few years. Applications include the testing of gas turbines, automobile engines, automatic testing of cables, strain testing of roads and bridges, quality control applications like life testing of batteries, environmental monitoring in research laboratories, using the PPM series 8000 or 8100 scanners. These are versatile measuring and controlling systems which can accept many different types of input, digital or analogue.

Each scanner has up to 100 channels and as each PET can operate with a number of scanners it is possible to process a very great amount of information — and the PET in its turn, can be interfaced with a main-frame computer as part of a wide-ranging local network.

A specific production line application of the use of PETs and the series 8100 scanner which PPM have installed is on a coffee packing production line. This set-up both counts the number of jars, and weighs the contents via electronic scales. Jars which are outside of control limits may be diverted, and the PETs print out a record of the maximum and minimum weights plus the average weight over a specified number of jars.

Many of the programmes written by PPM for their special applications (and there are no off-

the-shelf programmes for real time control applications) are very long, using up to 90K of memory, and as the PETs have only 32K, they were among the first companies in the U.K. to use 'overlaying' techniques with PETs.

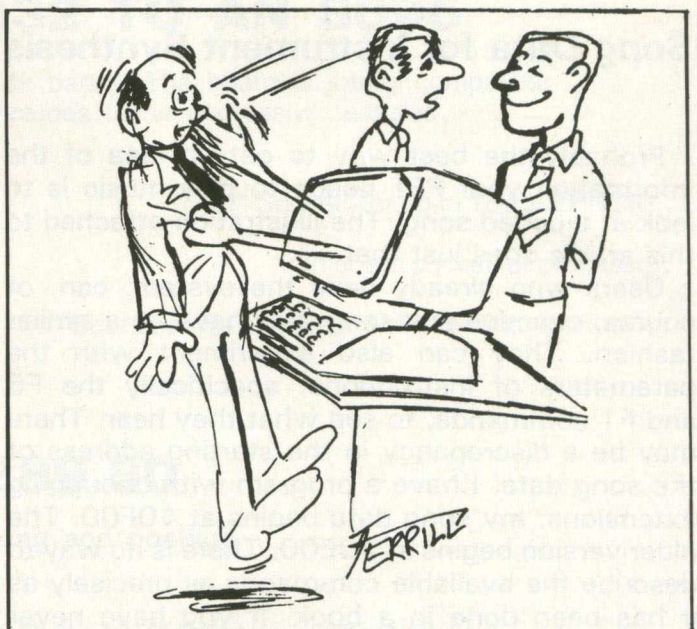
There are a number of other programmes for PPM PET-based systems — the Compucal Series 1 to 4 for instrument calibration and Compucal 5 for oscilloscope calibration, using appropriate test equipment for the purpose, along with the PET computer, disc drive and dot matrix printer.

The Compucal series are designed to simplify calibration techniques and give automatic print-out of test results with a wide range of instruments.

The oscilloscope programme also prints out results and it operates with oscilloscopes up to 500MHz band width.

Compucal is simple to use, and because it stores previous test results 1n floppy disc, these are available for comparison as required.

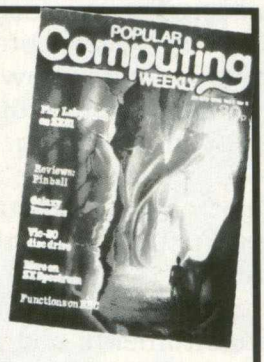
Many of PPM's installations are concerned with temperature control, environmental conditions, process monitoring or test procedures on everything from a nuclear power station to an electrical appliance for the home, so their systems are demonstrably adaptable and take the Commodore PET a long way from its usual applications as a desk-top computer concerned with working out the payroll. It too is adaptable.



'I Think a Simple Error Message Would Suffice.'

POPULAR Computing WEEKLY

HOBHOUSE COURT, 19 WHITCOMB STREET WC2



Every week for 30p

Available at your newsagent — NOW

Competitions
To tease and squeeze the best out of your rapidly developing computer brain.

Programming
To help you build your knowledge of sophisticated techniques. Step by step the secrets of the pros will be unfolded.

Clubs
What's happening inside Britain's mushrooming micro club world. Interviews and profiles of the new computing breed.

News
A really fast up-to-the-minute analysis of all that's going on in the expanding world of micro technology.

Reviews
A panel of experts assess the latest products both soft and hard as and when they become available.

Programs
Readers' own programs published each week with prizes for the best entries.

Questions
Answered by an expert who knows . . .
In short, it's a package you can't afford to be without. Make sure, place an order with your newsagent.

Sound and Vision

Song Data for Instrument Synthesis

Probably the best way to get an idea of the information your PET needs to play music is to look at a coded song. The illustration attached to this article does just that.

Users who already have the system, can, of course, examine any song they have, in a similar fashion. They can also experiment with the parameters of instructions, specifically the F6 and F1 commands, to see what they hear. There may be a discrepancy in the starting address of the song data. I have a program with debugging extensions; my song data begins at $0F00. The older version begins at $0E00. There is no way to describe the available commands as precisely as it has been done in a book. If you have never looked at a coded song, I'd recommend you do it with the manual on hand.

I have placed signposts on the code to help you see what commands are used and what they do. All commands are of the $Ex and $Fx type. You can see them in the first two sections of the code. In the third section, the flagged bytes mark the beginning of a musical "event" which loosely corresponds to a note. Note that these are excerpts. They do show several notes, as well as some of the structure of the song, but there is no continuity in the listing.

The music program was written in machine code by Dr. Frank Covitz. The program intercepts your instructions (mostly Ex and Fx commands followed by their parameters). It issues an error message if they don't make sense, or goes on to create instruments and play a song, if they do.

Just as in BASIC, precise syntax has to be observed. The syntax is described in the book. It consists mostly of how many parameters a command is to contain and what information is to be put into those parameters.

The code the user needs to enter consists of three major groups of communication:

1. decide what sort of sounds you would like to hear
2. build instruments and decide on how to play the song
3. provide the score to be performed by the PET.

The score itself is the simplest thing to enter. Music, its notes and durations are transcribed into numbers. That's section 3 in the illustration. Coding is easy, once you enter a measure of two of music, it's just tedious. Mistakes are costly, as room may need to be opened up or closed. Supermon or Extramon are handy; the "human interface," in the form of an editor, will be very helpful, indeed, when it's written.

The second major group is fun to write and is easy to do. That's part 1 in the illustration. It's the key to the whole system. It creates a base of numerous waveforms for each harmonic. You can copy the F5 instructions from the book, from songs you already have or you can invent your own by tranferring coordinates of your graph into the F5 command. Needless to say, if you have a way to analyze sounds, you can synthesize them back by entering a larger amount of data into the F5 parameters than shown here. That's apparently what Frank Covitz does when he synthesizes real sounds (wolf, oink-oink, bang!, no listen . . . dripping water, flutes, trombones and a genuine (?) 17th century harpsichord).

Finally, section 2 completes the requirements for a coded song. It consists of two parts.

The first part builds sound configurations (loosely called "instruments") from the waves calculated in section 1. This is where most of the fun is and most experiments can be performed. There is no limit to a variety of sounds you can create from a finite set of waves. You do it by varying the last three bytes of the F6 command, with the largest and most interesting impact achieved by variations in the third byte from the end. This byte controls the variable speed at which waves are scanned during sounding of a note. F6 command used after the F5 command is the powerhouse of the program. This is where you achieve the distinct sounds of the instruments, or families of instruments, you plan to use. It's really worthwhile to spend some time fooling around with the F6 command in the already coded song and listening to results. I can't think of a better way to learn the system.

The second part tells the interpreter the structure of the song. It includes addresses of the pieces, the order in which they are to be played, etc. as well as how to play a segment: its tempo, pitch offsets from the coded values, instrument assignments to voices, and stereo voice assignments (if you have two DACs). This information can be given once and will then be used for the entire song. Alternatively, it can be given for each piece or a group of song segments. It's your ballgame. You divide the song into meaningful segments based on the musical needs you detect (sound characteristics, timing changes, special dynamic changes, etc.) and you tell the PET how to play each segment.

* STRIKE NOW !!!! *

UPDATE YOUR 8032 TO AN 8096

with 64k of dynamic RAM arranged in 4 x 16k banks. The board is totally compatible with CBM 8096 memory boards and includes these impressive features :-

* Screen I/O peek through as on 8096.
* On board regulator for power supply.
* Straight forward simple installation.
* Sockets for up to 16k of EPROM. .
* On board socket for processor.
* Board comes complete with user documentation.

Price £289.80

including VAT packing and postage

Please send your cheque payable to Typecraft Systems (Derby) Ltd.,

to 128 Derby Road, Long Eaton, Nottingham. NG10 4ER.

Enquiries welcome - telephone 06076 69324

T.A.L.
COMPUTER DIVISION

AN INTELLIGENT CHIP

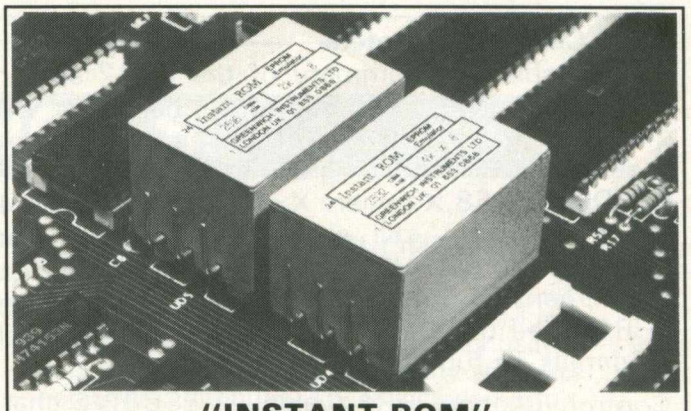
D.O.S. commands as per Universal Wedge program, and a sequential file reader.

?INTELLIGENT CHIP? Yes. The same chip will work in $9000, $A000 and $B000 locations with Pet Basic 2 and up.

£20 C.W.O.

11 High Street, Leighton Buzzard, Beds.
Tel: (0525) 372114

STOP PRESS: Amateur radio operators. R.T.T.Y. plug-in module for Pets and Vics. S.A.E. for further details.



"INSTANT ROM"

"Instant Rom" ROM/EPROM EMULATORS contain CMOS RAM with internal battery backup. When the power is switched off, data is retained for up to 10 years.
In the PET, a 4K INSTANT ROM can be fitted in the $9000 or $A000 socket. Machine-code (and Basic) programs can be stored, and are available at switch-on.
INSTANT ROM saves time. It can be used for long periods; when the program is finally "bug-free", an EPROM can be programmed.

4K INSTANT ROM (ROM socket replacement).....£56.00
2K INSTANT ROM (character generator replacement).....£39.00
Adaptor GA1 (essential for PET users).....£6.00

"G-ROM E"

G-ROM E is a 4K EPROM which will Auto-run, at switch-on, any Basic or Machine-Code program stored in INSTANT ROM. Basic programs can be stored with a few quick key-strokes. No skill is needed. Programs can now be run without a tape or disk unit, and can be changed without cost to the user. Diagnostic aids are included.

G-ROM E (specify type of PET).....£25.00

Postage (£1.00) and VAT are extra. Leaflets are available.

"INSTANT ROM" and "PETCLOCK" are COMMODORE APPROVED PRODUCTS.
GREENWICH INSTRUMENTS LIMITED, 22 BARDSLEY LANE, GREENWICH, LONDON SE10 9RF, UK. Tel: 01-853 0868. Telex: 896691 Attn. GIL.

Sound and Vision

```
C*
    PC  IRQ  SR AC XR YR SP          Meaning of the Code                Purpose
.; C6C3 9053 4D 00 3A 00 FA
.
.: 0F00 EE 10 F5 01 11.01.00 00. - EE manages memory.                   1
.: 0F08 02 D8.10 00.FF.02.00 00.   (2)F5 calculates $31 pages of waveforms Build base
.: 0F10 04 60.08 60.10 00.FF.00.   from which various instruments can be for sounds
.: 0F18 F5 12 31.01.00 84.1F 00.   built.
.: 0F20 FF.03.00 44.14 00.FF.04.
.: 0F28 00 38.10 00.FF.05.00 28.
.: 0F30 12 00.FF.07.00 20.10 00.
.: 0F38 FF.08.00 08.04 00.FF.00.
.: 0F40 F0 04 F6 01.1A.12 31 F6 - F0 sets 04 voices.                    2
.: 0F48 02.00.01 11 F6 73.11.01    (4)F6 build 4 instruments.           Build
.: 0F50 0C.00.0D 11 F6 14.00.01    (3)F1 assign instruments to voices left to instruments
.: 0F58 02.00.07 11 F1 C1.21 11    right.                               & song
.: 0F60 F1 C2.41 11 F1 C3.31 11    F2 sets tempo to $B8.                structure:
.: 0F68 EA.EA.EA.EA.EA.EA F2 B8    (n)FE play song segments coded at pp bb how to play
.: 0F70 FE 01 06 FE 01 BB FE 01    relative to $0F00.                   which part
                                                                        and when
.: 0F78 00 FE 01 06 FE 01 CB FE
.: 0F80 01 00 FE 01 EA FE 01 00
.: 0F88 FE 02 4F FE 01 00 FE 02
.: 0F90 64 FE 01 00 FE 02 14 FE
.: 0F98 03 6A FE 01 00 FE 01 EA
.: 0FA0 FE 01 00 FE 02 4F FE 01
.: 0FA8 00 FE 02 64 FE 01 00 FE
.: 0FB0 03 14 FE 03 75.EA.EA.EA.
.: 0FB8 F6 01.18.13 31 F2 5C FE - F6 changes sound of Ins#1.
.: 0FC0 03 90 FE 03 95 FE 04 C2    F2 redefines tempo to $5C.
.: 0FC8 FE 06 B6 FE 04 C2 FE 06    (n)FE play segments.
.: 0FD0 C1.EA.EA.EA F6 01.1A.12 - F6 again redefines Ins#1.
.: 0FD8 30 F2 4A FE 06 D6 FE 06    F2 changes tempo to $4A.
.: 0FE0 DC FE 08 13 FE 06 DC FE    (n)FE play segments.
.: 0FE8 08 23 FE 08 38 FE 08 38    00 in the last position ends command
.: 0FF0 FE 06 D0 FE 06 DC FE 08    string.
.: 0FF8 23 FE 08 38 00 00 00 00
.?
.
.: 1000 00 00 00 00 00 00 20 74 - *The music has been transcribed into 5 3
.: 1008 00 00 55 20 70 00 00 15    byte events.                         What to play
.: 1010 10 30 00 61 00 10 74 00    The first byte is duration of a note.
.?                                 Other four are notes translated into
.                                  numeric code.
.: 10A8 00 00 00 30 F2 6D 00 5D  *The score has been cut into musically
.: 10B0 0E 74 00 00 1D 02 00 00    meaningful segments. Only excerpts are
.: 10B8 00 00 00 20 F4 6C 63 5C    shown here. A segment ends with a 00 in
.: 10C0 20 34 00 00 1C 1E 34 00    the duration position. The first segment
.?                                 begins at $1006, ends at $10BA. The
.                                  second begins at $10BB. Last segment
.: 1720 2F 53 00 20 EB 00 00 53    begins at $1738, ends at $18AE. A zero
.: 1728 23 2B 00 5A 13 26 2B 63    in $18AF and in the command string at
.: 1730 1A 13 30 2B 23 1A 13 00    $0FFC terminates the song. Segment
.: 1738 10 00 00 00 00 10 77 00    addresses in FE commands are relative
.: 1740 00 00 20 79 00 00 00 30    to $0F00.■
.?
.
.: 1838 2A 23 17 22 73 2A 23 17
.: 18A0 44 F4 00 00 5C 26 34 00
.: 18A8 63 1C 38 34 68 00 1C 00
.: 18B0 00 00 00 00 00 00 00 00
.?
.
READY
```

8032 Control Characters

This table is a summary of the 8032 screen control functions. The ESC/RVS characters will display as lower/upper case or upper case/graphics, depending on which mode you're in. POKE59468,X (where X = 12 for graphics, 14 for lower case) still changes modes without changing the gap between the lines. Notice that complimentary functions differ by 128 using CHR$). See the Commodore BASIC 4.0 manual for details on functions.

Control Function	CHR$ (value)	ESC/RVS char.
BELL	7	g
GRAPHICS	142	shift n
TEXT	14	n
SCROLL DOWN	153	shift y
SCROLL UP	25	y
SET BOTTOM	143	shift o
SET TOP	15	o
INSERT LINE	149	shift u
DELETE LINE	21	u
ERASE BEGIN	150	shift v
ERASE END	22	v
SET/CLR TAB	137	shift i
TAB	9	i

The above describes the special 80 column screen control functions. The functions can be activated two ways; by using CHR$ (and the appropriate value or, preferably, by placing the appropriate character in reverse field within quotes. This is done by entering quote mode, hitting 'ESC', then 'RVS' and the character. For example, to do a Scroll Down enter quote mode and type 'ESC', 'RVS', shift & 'Y' and RETURN. 'ESC' takes you out of quote mode. If you wish to continue with more characters following the Scroll Down you'll have to do an OFF/RVS, another quote and DELete the quote. This is comparable to the cursor control characters but not quite so automatic.

Although you could use the CHR$ (values, the ESC/RVS method saves bytes and will eventually become much more legible. After all, when was the last time you used a CHR$ (17) to do a cursor right. (or is it a cursor up? . . . or is 17 delete? . . . no, I think it's cursor down . . . I'd better check . . . hmm).

There is still another way to activate these functions without using PRINT. This is directly from the keyboard. But you say "There is no key on the keyboard assigned to do scroll down or set top . . .". By pressing certain combinations simultaneously, the keyboard value that is passed to the operating system will be the CHR$ value that activates the function. Many combinations do the same functions. I've listed only the easiest ones to remember.

CONTROL FUNCTION	KEY COMBINATION
TEXT	WHO KNOWS?!
GRAPHICS	BOTH SHIFT/"
SCROLL DOWN	LEFTSHIFT/TAB/I
SCROLL UP	PVS/TAB/I
SET BOTTOM	Shift/z/A/L
SET TOP	z/A/L
INSERT LINE	Shift/RVS/A/L
DELETE LINE	RVS/A/L
ERASE BEGIN	Shift/TAB/leftarrow/DEL
ERASE END	/TAB/leftarrow/DEL
SET/CLR TAB	Shift/TAB
TAB	TAB

The space beside TEXT is empty because it hasn't been found yet. If anyone does, please let me know.

The window can also be POKEd to size. The pokes are:
Screen TOP: 224,T where T = 0 to 24
BOTTOM: 225,B where B = T to 24
LEFT: 226,L where L = 0 to 79
RIGHT: 213,R where R = L to 79

I'm not sure what weird or interesting effects you can get by making TOP less than BOTTOM or LEFT greater than RIGHT. Try it and see!

Give your PET a home.... Buy it a PETDESK!

A Commodore approved product.

Specially designed to take any Commodore Pet system.

Black leathercloth top and Black metal frame. Paper feed tray, top extension shelf. Concealed cables and 4 way 13 amp plug socket. Mounted on castors. Size 1470 x 560 x 675 mm. Delivered flat packed.

Price £189.50 includes VAT and delivery.

This offer available UK only. Cheques with order to:
Tirith Ltd, Pear Tree House, Woughton on the Green, Milton Keynes MK6 3BE. Telephone: (0908) 679528

EPSON TYPE 3 PRINTERS

Epson's superb Type 3 series replaces the Type 2 and offers:
* IMPROVED PERFORMANCE
* MANY NEW FEATURES
* LOWER PRICES

For free brochure and special Epson offer write to Datatech Ltd, stating your computer.

MINI DISKS — BIG SAVINGS

VERBATIM DATALIFE 5¼"

MD525 S/Sided 40 Track £16.45
MD550 D/Sided 40 Track £22.45
MD577 S/Sided 77 Track £24.45
MD557 D/Sided 77 Track £32.45

Prices are per ten pack, excluding VAT.

All disks are factory fresh, soft-sectored, hub reinforced and certified 100% error-free.

Please include UK p&p at 95p per pack, then add 15% VAT to total.

Send cheque/PO now to Datatech Ltd.

DATATECH LTD, DEPT CC, FREEPOST, ALTRINCHAM, CHESHIRE, WA15 0BR.

VACANCIES

ENGINEER

Computer Services Midlands is one of the Nation's leading Commodore dealers in both computer sales and software development. A requirement has now arisen for a further engineer to operate in the East Midlands area maintaining the hardware of our many Commodore users. Preference will be given to an experienced engineer familiar with the Commodore range but full training is available by means of both manufacturer's courses and our own engineering staff.

An attractive salary based upon previous experience is offered together with company car and the chance to join the planned expansion of a young exciting organisation.

Ask for Mike Ward when writing or telephoning.

SOFTWARE DEVELOPMENT

Our business software is already regarded by many as the finest on the Commodore range and with new products being continually added our requirement for quality software personnel continues. These include both junior and senior programmers and a department manager responsible for the efficient running of the department. Excellent salaries are offered in all cases together with a company car for the most senior position.

Contact Iain Caldwell or Peter Mart for further details.

CSM Computer Services Midlands

Computer Services Midlands Limited, First Floor, Refuge Assurance House, Sutton New Road, Erdington, Birmingham. B23 6QX. Tel: 021-382 4171.

Programming Tips

Getting to Grips with Machine Code

Assembly Language

You may have thought that programming in machine language via the Monitor could really be a chore if you had to write a complete application. And you would have been correct, but at least we have a Monitor which conveniently allows us to type in a hex code at a certain memory location. Before our programming time, this wasn't possible. On early machines, programmers had 8 or 16 (or 32) toggle switches which could be set in one of two positions — on or off. To load an instruction, for instance a hex "60" (return), you would first have to set the eight toggle switches in the binary representation of 60, then push a "load" button to load the code into the machine. Our simple program from the last article might take an hour to load! But let's not dwell on the past, since higher level "english-like" languages have been developed for our use.

Assembly Language developed because even though sophisticated monitors and keyboards were developed, it still was a chore to remember and code in actual hex (or binary or octal) instructions, or look at a "hex dump" of a program and be able to see what the program did. (I do understand though, that programmers still exist who can do this at will, but they are kept chained inside the old ENIAC computer). For the same reasons that higher level languages such as BASIC began to replace Assembler programming, Assemblers were developed to replace actual Machine Language programming.

What is an Assembler? An Assembler is basically a program (which can be written in any language, even BASIC) that reads in a source file that contains lines of mnemonic instructions and "assembles" each mnemonic instruction into its Machine Language equivalent. The nice feature about Assembly Language is that instead of having to remember all of the various hex forms of an instruction, you only have to remember one mnemonic. For example, the instruction to "Load the Accumulator" can be represented by "A9" in the immediate mode, "AD" in the absolute mode, "A5" in the zero page mode, and so on. In Assembly Language, the instruction is merely represented as "LDA" and the Assembler resolves which form the source line is representing. So you can see that writing in Assembly Language is much easier than hand-coding in Machine Language, yet it yields the same benefits.

If you turn now to the listing of the program accompanying this article, you will first notice that it is broken down into four distinct columns. The first column represents a line number which is arbitrarily assigned for reference purposes; the next column is the actual starting memory location at which this line of code will be loaded; the next column contains the actual hex code generated for that line; and finally, the actual Assembly Language source text.

(If ever you should read about or come across a program written in Assembly Language that might be useful to you, and the author provides a listing such as the one with this article, then all you need do to implement it is to type in the code and assemble it. If you don't have an Assembler, then simply go into the Monitor and begin typing in the code as listed in the third column, starting at the memory locations listed in the second column. When finished, do a Machine language "save" and you will have a copy of the program).

Assembly Language Format

Without actually worrying about what exactly this program does, let's look at the program listing from a syntactical point of view. The first rule you will notice about this Assembler (Commodore's Assembler Development System) is that a comment line begins with a semicolon (;). In my listings, I generally put a lot of comments, and since there is no overhead associated with it, I feel it is a good practice for you to do as you develop an Assembly Language program.

The next portion of the program (lines 24-28) contains examples of the Assembly Language assignment statement. This statement allows you to assign a value to a variable (or label). If you precede the value with a "$", then the Assembler assumes a hex value, otherwise it assumes decimal (this is true whenever referring to values throughout your programs). Hexidecimal is usually more convenient because most memory maps and other listings are in this format. By using the assignment statement (in the beginning of a program ONLY), you tell the Assembler to use the associated value every time it encounters that label in the program. For instance, line 30 assigns the hex value "0070" to the variable "chrget". If you look at the actual code that references this variable (line 47) you can see the purpose of the assignment statement is: 1) to make your program more meaningful and

easier to read; and 2) to make it easier to change (if the value for "chrget" changes in the future). In general, assignment statements are used to refer to the memory locations of routines (or subroutines) that are either part of the operating system (as in this case), or some of your own routines.

The next part of the program, lines 47-50, is the main routine which basically executes three subroutines. Notice that instead of specifying an exact location to "jump to," I have used the labels that were defined in the beginning. The label "aschex" is not listed in the beginning of the routine because it is actually part of this program, at line 57. Notice that the Assembler generated the address of "0344" for this label because, as line 57 specifies, the memory location that the routine or label "aschex" would begin at is "0344."

As you can see by now, labels within an Assembly Language program work similarly to their counterparts in BASIC and other high level languages. So one obvious advantage Assembly Language has over Machine Language is the fact that all "jump to" and "branch to" (as in line 61) addresses can be resolved for you by the Assembler. The next part of the program, lines 57-82, is the actual routine which we will look at later in more detail.

The final part of the program at line 88 illustrates what is known as an Assembler directive. An Assembler directive is basically a command that tells the Assembler to perform some operation at "assemble time," but it is not a part of the actual program. In this Assembler, directives must be preceded by a period (.). The "byte" directive in line 88 tells the Assembler to set aside one memory location, and in this case, put a hex "0" there. In this instance, the memory location used is "0371." Notice that all the code that references the label "answer" (lines 59, 71, 76, 79, 80) contains the address "0371."

More Instructions to Work With

Let me first give you some background on the accompanying article. If you have already paged through this issue, you may have noticed another article which describes a routine that allows you to program the cursor directly from BASIC, using X,Y coordinates. For example, to move the cursor to line 15 and column 40, you could type "sys address,15,40". The "sys" command cannot directly pass the parameters "15" and "40", so one of the functions of the "sys routine" must be to accept these parameters.

Sound easy so far, right? Well yes, except that when you code "sys address,15,40", the number "15" is actually represented as two characters in memory — hex 31 (for "1") and hex 35 (for "5"). This is what is known as the ASCII character representation. If we want to be able to use the number "15" as some sort of counter (of the number of times to cursor down), then we must convert the ASCII representation of "15" to the actual numeric value. We will look here at the subroutine that does this.

Still sounds easy, Right? Yes, except that this explanation overlooks one thing — how do we pull the number "1" and the number "5" from our BASIC statement. Well I won't explain in detail how this works in this article, except to tell you that your PET/CBM already comes with a routine which will do this. This routine is referenced in my listing by the label "chrget", and when it is invoked (via the "jsr"), the next character or token from the current BASIC line will be loaded into the Accumulator. And the pointer to the current character will be incremented so that it now points at the next character in the line.

Now let's look at the program.

```
1:aschex.src.....page 0001

line# loc   code        line

00001 0000              ;*****************************************************************
00002 0000              ;****                                                         ****
00003 0000              ;**** program name = aschex.src                              ****
00004 0000              ;**** author = dave scott. cbm  us                           ****
00005 0000              ;**** date = 03/03/82                                        ****
00006 0000              ;****                                                         ****
00007 0000              ;**** purpose: the purpose of this program is to convert a two digit ****
00008 0000              ;****          ascii number to hex.  the routine is entered via the  ****
00009 0000              ;****          "sys" command and the two digit number should follow  ****
00010 0000              ;****          the sys command as in the following example:          ****
00011 0000              ;****                                                         ****
00012 0000              ;****          sys 826,25                                     ****
00013 0000              ;****                                                         ****
00014 0000              ;****          where 826 is the sys address and 25 is the number to  ****
00015 0000              ;****          convert.  the converted number will be stored at      ****
00016 0000              ;****          location 0371 hex.                             ****
00017 0000              ;****                                                         ****
00018 0000              ;*****************************************************************
00019 0000              ;
00020 0000              ;*****************************************
00021 0000              ;****      kernal routines        ****
00022 0000              ;*****************************************
00023 0000              ;
00024 0000              outchr=$ffd2            ;print char in acc
00025 0000              ;
00026 0000              ;*****************************************
00027 0000              ;****    rom and system routines     ****
00028 0000              ;*****************************************
00029 0000              ;
00030 0000              chrget=$0070            ;char get (in 4.0)
00031 0000              chrgot=$0076            ;char got (in 4.0)
00032 0000              sunerr=$bf00            ;syntax err (in 4.0)
00033 0000              ;
00034 0000              ;*****************************************
00035 0000              ;****    system storage areas       ****
00036 0000              ;*****************************************
00037 0000              ;
00038 0000              txtptr=$0077            ;current char
00039 0000              ;
00040 0000              ;*****************************************
00041 0000              ;****       main routine            ****
00042 0000              ;*****************************************
00043 0000              ;
00044 0000                      *=$033a
00045 033a              ;
00046 033a              ;
00047 033a  20 70 00            jsr chrget                      ;get 1st digit
00048 033d  20 44 03            jsr aschex                      ;convert number
00049 0340  20 70 00            jsr chrget                      ;get char before
00050 0343  60                  rts                             ; return to basic
00051 0344              ;
00052 0344              ;*****************************************
00053 0344              ;***  convert number from ascii to hex  ****
00054 0344              ;*****************************************
00055 0344              ;
```

```
1:aschex.src.....page 0002

line# loc   code        line

00056 0344              ;
00057 0344  38          aschex sec                              ;set carry for sub
00058 0345  e9 30               sbc #48                         ;1st digit to hex
00059 0347  8d 71 03            sta answer                      ;store it in answer
00060 034a  20 70 00            jsr chrget                      ;get second digit
00061 034d  90 07               bcc asc1                        ;goto to asc1 if
00062 034f              ;                                         btwn 0-9
00063 034f  c9 2c               cmp #44                         ;   is it = ","
00064 0351  f0 1d               beq ascend                      ;     yes. done
00065 0353  4c 00 bf            jmp synerr                      ;     no. syntax error
00066 0356              ;
00067 0356  38          asc1    sec                             ;set carry for sub
00068 0357  e9 30               sbc #48                         ;2nd digit to hex
00069 0359  48                  pha                             ;save 2nd digit
00070 035a  a9 00               lda #0                          ;
00071 035c  ae 71 03            ldx answer                      ;find tens by mult
00072 035f  18          asc2    clc                             ;
00073 0360  69 0a               adc #10                         ; 1st x 10
00074 0362  ca                  dex                             ;
00075 0363  d0 fa               bne asc2                        ;
00076 0365  8d 71 03            sta answer                      ;store tens in temp
00077 0368  68                  pla                             ;get 2nd digit
```

Programming Tips

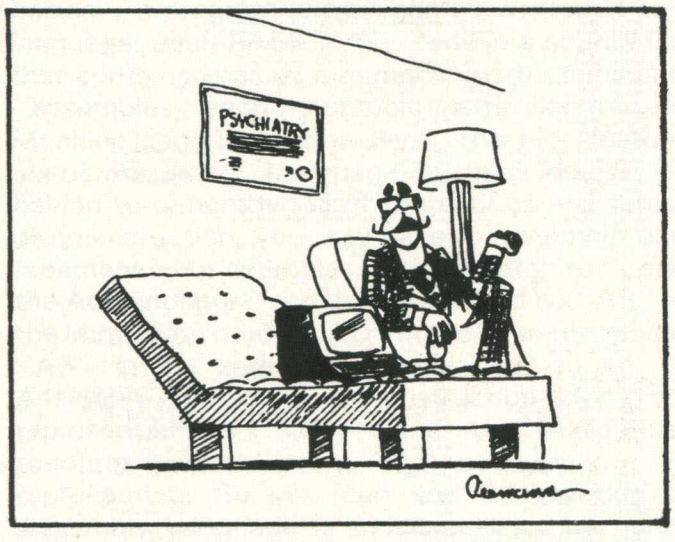
```
00078  0369  18              clc                             ;
00079  036a  6d 71 03        adc answer                      ;  and add to temp
00080  036d  8d 71 03        sta answer                      ;  save it in temp
00081  0370                  ;
00082  0370  60              ascend rts                      ;done
00083  0371                  ;
00084  0371                  ;*****************************************
00085  0371                  ;****      storage areas          ****
00086  0371                  ;*****************************************
00087  0371                  ;
00088  0371  00              answer .byte 0                  ;hex result
00089  0372                  ;
```

Lines 47-50

This series of lines is the main routine of this program. When the "sys" command is encountered in your BASIC program or from immediate mode, control is passed to the routine starting at the memory address following the "sys." In this case, the address is 826 or hex 033a. The first thing the routine does is get the next character from the BASIC line. (The format for the "sys" parameter is: sys 826,x. When the routine is first entered, the character in the Accumulator is a ",".) This routine will then load the Accumulator with the first (and maybe the last, if it is a single-digit number) digit of the number.

Line 48 will then jump to the subroutine that converts the ASCII representation of the number to hexidecimal. Note that the first digit of the number is sitting in the Accumulator. We will explain the actual routine in a minute.

Line 49 will jump to the subroutine that gets the next character in the line. At this point, the routine has already extracted the final digit (if there was a second digit) and converted the number passed by the "sys" command. Calling this subroutine and its parameters. We do this because the BASIC interpreter will resume its operation when the return in Line 50 is executed. The character in the Accumulator must, at this point, be the one following our "sys" command (perhaps a ":").



'How About Kicking Off With a Good, Old-Fashioned Memory Dump?'

Lines 57-59

This set of lines introduces us to three new instructions. The first instruction "sec" will "Set the Carry Flag." This is done because of the nature of the "sbc" or Subtract With Carry" instruction. The "sbc" instruction will subtract the value following it from the value in the Accumulator, placing the result in the Accumulator. If the resultant value is greater than or equal to 0, then the Carry Flag will be set. If the result is less than 0 (i.e., if the value in the Accumulator was less than the value being subtracted), the Carry Flag is cleared. Because of the nature of binary arithmetic (two's complement form), it is required that the Carry Flag be set when subtracting single precision numbers. For a more detailed explanation of the Subtract instruction, refer to the MOS Programming Manual or some other suitable 6502 reference guide.

Getting back to our subtraction, remember that the Accumulator already contains the ASCII representation of the first digit (or "tens" column digit for a 2-digit number). To find the actual numeric value of this digit, we must subtract the ASCII value of the character "0" from the ASCII value of the digit. The easiest way to do this is to use the immediate form of the Subtract instruction, which is represented by the "=". (As was mentioned before, the Assembler will assume a decimal value unless specified). For instance, the number "2" would be represented as "50" or in hex as "32". By subtracting ASCII "0" which is actually "48" or in hex, "30", you can see that the resultant value will be "2" (50 – 48 = 2). So after preforming the instructions on Line 57 and 58, the Accumulator contains the actual numeric value of the first digit.

Line 59 introduces us to the "sta" or "Store the Accumulator in Memory" instruction. This command simply stores a copy of the value in the Accumulator at a certain memory location. This case illustrates the absolute form of the instruction in that the label following is an absolute location in memory. Remember that we defined the label "answer" at the very bottom of

the program, so the value will be placed at location hex 0371.

Lines 60-65

Line 60 first gets the next character from the line and stores it in the Accumulator. This character can be the second digit of the number, a comma (if it is a one-digit number), or some other BASIC token.

If the Carry Flag is not set (by routine "chrget") then we know that the character in the Accumulator is the ASCII representation of a number between 0 and 9. The "bcc" or "Branch on Carry Clear" instruction on Line 61 is another type of branch instruction that will branch to another point in the routine (in this case, "asc1", if the Carry is not set). If there is a second digit, this path will be followed.

If the character is not a number between 0 and 9, then we want to check to make sure it is a comma. Note that this check is up to you as a programmer (i.e., I have decided that one digit numbers must be followed by a comma). The "cmp" or "Compare the Accumulator with Memory" instruction on Line 63 compares the value now residing in the Accumulator with the immediate value of "44" ("44" is the ASCII representation of a comma). The "cmp" instruction actually subtracts the value in memory (in this case, the immediate value "44") from the value in the Accumulator, but DOES NOT store the result anywhere. Depending on what the result of the operation was, one of three Status Flags will be set. The Negative Flag is set if the result is less than zero, the Carry Flag is set if the result is greater than or equal to zero, and the Zero Flag is set if the result is equal to zero.

The "beq" or "Branch on Result equal to Zero" instruction in Line 64 will cause a branch to the end of the routine, "ascend" (Line 82), if the present character is a comma. If this is the case, the "sys" command only passed a one-digit number. If the character is anything else, the routine will jump to a routine that prints "syntax error". This is done on Line 65.

Line 67-69

If the character in the Accumulator is a number between 0 and 9, indicating a second digit, then control continues at Line 67. Lines 67 and 68 will again take the ASCII representation of the second digit, and subtract an ASCII "0" from it. The resultant value in the Accumulator will then be the actual numeric representation of the second digit.

We have now converted the two digits down to their actual numeric forms. The first digit at this point is stored at the memory location labeled "answer" and the second digit is in the Accumulator. Now what? Well I now have to disturb the cobwebs of your mind by referring to some concepts that were taught in elementary school — those concerning the "tens" column and "units" column of numbers. To convert this number, we must first multiply the "tens" column by 10, then add the "units" column to that answer.

But before going on to the instructions that do this, we first must save the second digit. I do this in Line 69 with the "pha" or "Push the Accumulator onto the Stack" instruction. What's a Stack? The easiest way to explain a stack is to use the analogy of a "plate stacker" used in many restaurants. As clean dishes are brought out, they are placed in the stacker. When a dish is needed, it is taken off the top and the stacker pushes upward. The idea is that the most recently placed dishes are the first to be taken off. The Stack in the PET/CBM works the same way. The values placed on the Stack are the first to be taken off. The only thing that you must be careful of is to make sure that values are taken off in the proper order.

Lines 70-77

How do we perform a multiplication? The way I've chosen here is a technique known as the "successive addition" method. In other words, we start with 0 and add 10 to the result of the previous addition successively for the number of times specified by the "tens" column digit. For example, if our first digit is "3", we perform the operation, 0 + 10(1st) + 10(2nd) + 10(3rd), which gives us 30.

To implement this, we first perform the instruction "lda" or "Load the Accumulator with the value in Memory" as in Line 70. Again we have the immediate form of the instruction which in this case loads the value "0" into the Accumulator. Next we load the X register with the value of the "tens" column to be used as the counter for the upcoming loop.

We next enter the loop which starts at Line 72 at label "asc2." First, we set up for an addition operation by performing a "clc" or "Clear the Carry Flag" instruction as on Line 72. This instruction resets the Carry Flag. We do this because the next instruction, the "adc" or "Add Memory to the Accumulator with Carry", will use the Carry Bit (if it is set) as part of the addition. In Line 73, we add the immediate value 10 to the contents of the Accumulator (which at this point

is 0). The resultant value will be stored in the Accumulator. As in other operations, certain status flags will be set according to the outcome of the operation. For more information on this, refer to MOS Programming Manual.

Line 74 will decrement the "tens" column value which we initially stored in the X Register. This is done with the "dex" or "Decrement the X Register" instruction. Again, certain status flags will be set according to the outcome of the operation.

If the resultant value in the X Register after the "dex" instruction is performed is 0, the branch instruction on Line 75 will not be executed. If the value is still greater than 0, the loop will be entered again at Line 72. You will notice that the number of successive additions is controlled by the value of the X Register or the "tens" column digit. When we fall out of the loop at Line 75, the value in the Accumulator will be the final result of the successive additions. All we need to do now is add the "units" digit to the value in the Accumulator and the resultant value will be our converted number.

Line 76 stores the "tens" column calculation at location "answer." Note that we have already stored the first digit at this location in line 59, so this operation will overwrite that value. The reason for this is simple. If the number to convert was only a single digit, then the value in "answer" is the converted value when we branch to the end of the program in Line 64. If the number to convert is two digits, then the value in answer can be overwritten.

Remember that we originally stored the "units" digit on the Stack in Line 69. Now we need to pull it off, so we do this with the "pla" or "Pull top value off the Stack and place into the Accumulator" instruction. Remember that if you are not careful to pull values from the Stack in the order that they were placed there, you could get yourself in trouble. Notice also that the Stack is a handy place to store values.

Lines 78-82

At this point, we have the "tens" column calculation in location "answer" and the "units" column value in the Accumulator. All we have to do is add these two values together. Lines 78 and 79 do exactly that. After the calculation is performed, we store the result of the addition at storage location "answer", and finally return to the main routine at Line 82.

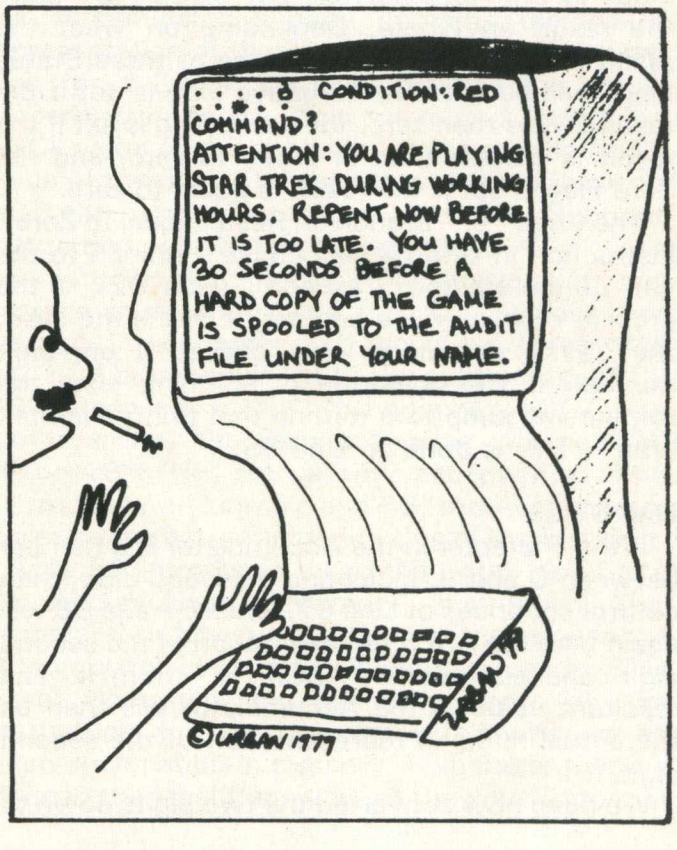
SUMMARY

To summarize, the program first gets the first digit from our BASIC line (Line 47). Once it has done this, it jumps to the conversion routine (Line 48). This routine first converts the first digit (Lines 57-59) and stores it at "answer." Then it gets the next character from the BASIC line (Line 60) and checks to see if it is numeric (Line 61), or if it a comma (Line 63). If it is numeric, then it branches to the routine that gets the second digit and converts the number. If it is a comma, then we know it is only a one-digit number and so the routine branches to the end (Line 64). If it is neither of these types of characters, then "syntax error" will be printed (Line 65).

If it is a two-digit number then first we convert it to its actual numeric representation (Lines 67-68). Then we set up to perform our successive addition using the "tens" column digit as the counter (Lines 69-71), do the actual addition (Lines 72-74), add the "units" value to that result (Lines 76-79), and finally, store the converted number at location "answer" (Line 80).

And that's it. Easy, right?? Have fun and experiment!

If you should have any questions or comments, please write to me via the Editor.



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, brain-teasers 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

Interfacing

Pet to Teletype Interface

The interface described below was received from Lt. W. Hawes in Nova Scotia. Note: it will operate with 8 level TTY's but not 5 level machines.

Interface Description

The cct. shown in Fig. 1 is a modification of an interface that was originally built to output to a TTY from the PET Parallel User Port. The problem with the Parallel port was that software was required to be resident in memory in order to output data and LISTing of programs was not possible since the operating system has control during a LIST. Clearly the way to go was from the IEEE 488 Port.

The modification to output from the IEEE Port was based on a cct. by Prentice Orwell. Some of the features of my original cct, such as UART vice shift register and closk frequency from PET vice interface oscillator, were retained.

My cct. is as shown in Fig. 1. It uses a + 5v and − 12v (originally only a dual supply UART was immediately available) for both the UART and the 20mA current loop. The cct. could be further simplified to a single + 5v supply as shown in Fig. 2 by using a single supply UART such as the AVA − 1014A or equivalent. The 20mA loop could then be constructed using spare inverters on the 4049's.

As stated above, hardware is reduced by omitting the interface oscillator. PET itself supplies the 1760 Hz (16 × baud rate) UART clock frequency from CB2 on the parallel port.

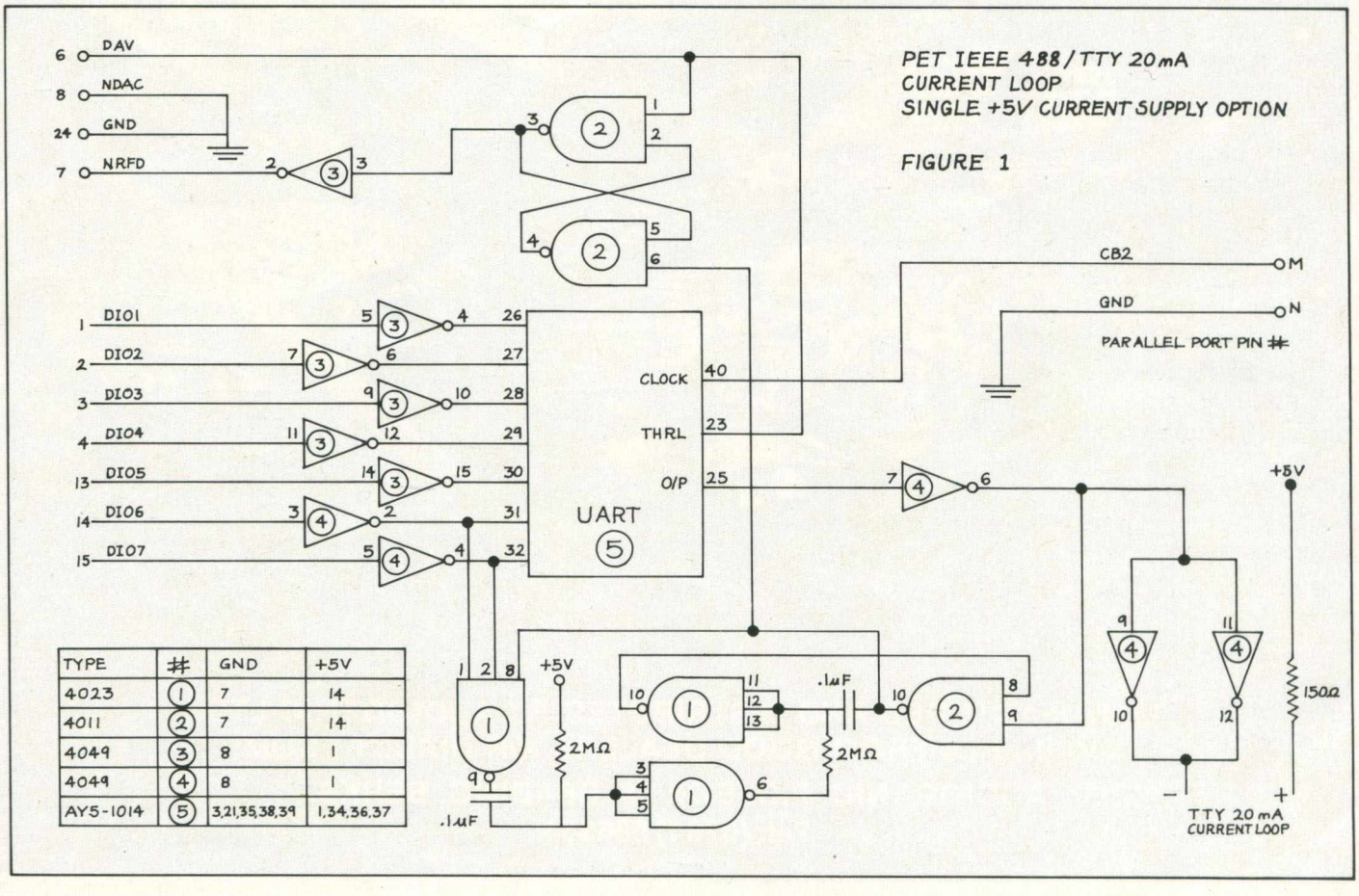
Circuit Operation

Initialize: POKE 59467,16: POKE 59464,69: POKE 59466,51 (outputs 1760 Hz from CB2 to UART, tape I/O disabled)

Operate: OPEN 4,4: CMD 4 (Printer primary output device — enter from keyboard to LIST or include in program to be RUN)

Return to Screen: PRINT 4 (from keyboard or include in program)

System Recover: POKE 59467,0 (restores correct tape I/O)



TYPE	#	GND	+5V
4023	1	7	14
4011	2	7	14
4049	3	8	1
4049	4	8	1
AY5-1014	5	3,21,35,38,39	1,34,36,37

Basic Programs

Weeny

Reference the Weeny Word Processor published in your April/May issue, the idea is to good to leave in that state (aren't they all!). Herewith an extension to allow disk I/O, and two utilities to improve the printing aspect. The program assumes 8000 series — sorry, but the window facility is used to avoid messing up the screen with 'ready''. etc (line 3000).

The essential philosophy is retained, i.e. character by character processing in BASIC, and use of the screen editor to prepare text (the 8032 is excellent for this — don't forget the TAB feature). Instructions for the original carry over, i.e. use SHIFT and RETURN not just RETURN to move to the next screen line; leave line 1 blank for control; end the text with back arrow; to save/recall/print a screen page, home the cursor and enter 'run' and the RETURN key. All control is strictly alternative technology level: disk files are overwritten automatically if they exist — no warning; no file-not-found check; etc.

When choosing 'edit' the program drops into BASIC, using the window as indicated above. To edit, first key HOME twice to reset the window.

The two utilities provide better print facilities. One merges screen pages on disk into print pages; the other prints direct from disk instead of from the screen. Hence (typically) the three screen pages that make a print page can be merged and then printed in one go, instead of recalling to screen and then printing each of the three parts separately. To exit either of the utilities key RETURN on any prompt, or just STOP.

Lastly, the mystery of lines 14, 15, 17. These will be plain sailing to owners of POWER (which should be all PET programmers). With POWER on, SHIFT X deletes the current cursor line with scroll up, SHIFT Y creates a blank line with scroll down, SHIFT Z deletes to end of line from current cursor position. They are just three of the POWER keys/routines I use regularly, hence the strange line numbers. To use X, Y and Z in text key INST first.

Yours
Dr Peter Hartley —
Stage One Computers Ltd
21.6.82

Not Quite So Weeny WP

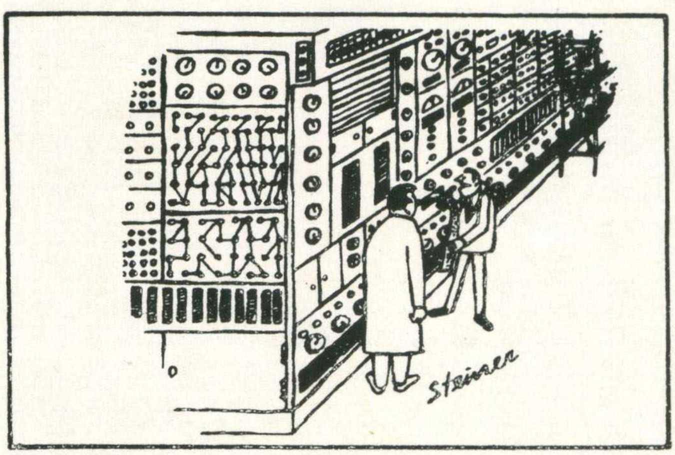
```
 14 rem"y=■
 15 rem"x=■
 17 rem"z=■
 100 rem weeny word processor - pjh version
 150 gosub1000:print"■■":ifq$="r"then2000
 200 open3,3:ifq$="p"thenopen4,4:print#4,"■";:goto300
 250 open4,8,4,"@"+d$+":"+p$+",s,w"
 300 get#3,a$:ifa$<>"■"then400
 310 ifq$="p"thenprint#4
 320 ifq$="s"thenprint#4,a$
 330 close4:close3:goto150
 400 print#4,a$;:ifa$=chr$(13)andq$="p"thenprint#4,"■";
 500 goto300
 1000 print"■■■s■ave or ■■■rint or ■■■ecall or ■■■dit";
 1100 getq$:ifq$<>"s"andq$<>"p"andq$<>"r"andq$<>"e"then1100
 1200 ifq$="s"orq$="r"theninput"■■Page Name";p$:input"■■Drive #";d$
 1250 ifq$="e"thengosub3000:end
 1300 return
 2000 print"■":dopen#1,(p$),d(val(d$))
 2100 get#1,a$:s=st:printa$;:ifs=0then2100
 2200 dclose#1:goto150
 3000 print"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■";:return
ready.
```

Merge Page Utility

```
 100 rem weeny word processor - merge screen pages
 150 gosub1000:goto2000
 1000 input"■■Destination Page Name";p$:input"■■Drive #";d$
 1100 dopen#1,(p$),d(val(d$)),w:ifds=0thenreturn
 1200 dclose#1:goto1000
 2000 input"■■Next Merge-Page Name ■■■";p$:input"■■Drive #";d$
 2030 ifp$="■"then3000
 2050 print"■■Merging       ":dopen#2,(p$),d(val(d$))
 2100 get#2,a$:s=st:ifa$<>"■"thenprint#1,a$;
 2150 ifs=0then2100
 2200 dclose#2:goto2000
 3000 dclose#1:goto150
ready.
```

Print From Disk Utility

```
 100 rem weeny word processor - direct print from file
 150 gosub1000:goto2000
 1000 input"■■Page Name";p$:input"■■Drive #";d$
 1300 return
 2000 print"■Printing Page":open4,4:dopen#1,(p$),d(val(d$))
 2100 get#1,a$:s=st:ifa$<>"■"thenprint#4,a$;
 2150 ifs=0then2100
 2200 dclose#1:close4:goto150
ready.
```



'It Wants a Squirt of Oil on that Squeaky Door Hinge Back There.'

Basic Programs

CAR RACE

```
10 DIM A1(14),B1(14),C1(14),GS(5),BS(10),MS(14)
20 X=32859:G=1:R=0:S=0
21 PRINT"[CLR]
30 FOR Z=1TO14:READ A,B,C:A1(Z)=A:B1(Z)=B:C1(Z)=C
40 FOR D=1TO A:POKE X,42:X=X+B:NEXT D
50 X=X+C
55 NEXT Z
60 DATA 20,1,40,3,41,-40,4,1,40,2,41,-1,2,40,-1,6,-1,-40,3,-41,40
62 DATA 2,39,1,2,39,-40,4,-1,-40,5,-41,40,12,-1,-40,2,-41,2,6,1,1
70 FOR Z=1TO5:READ GS(Z):NEXT
75 DATA 2,4,1.9,1.5,1.2,1
80 FOR Z=0TO9:READ BS(Z):NEXT
85 DATA 0,6,15,20,25,30,40,59,80,98
90 FOR Z=1TO14:READ MS(Z):NEXT
95 DATA 70,70,70,85,70,70,35,40,60,60,70,80,40,200
100 D=10:GOSUB 5000
105 PRINT"7 TO CHANGE UP
107 PRINT"4 TO CHANGE DOWN
110 PRINT"8 TO BRAKE
112 PRINT"9 TO ACCEL
114 PRINT"S TO START
116 GET S$:IF S$="" THEN 116
120 X=32859:Y=X:S9=TI
130 A=3:GOSUB 4000
200 FOR Z=1 TO 14
210 A2=A1(Z):B2=B1(Z):C2=C1(Z):MX=MS(Z)
220 D=15:GOSUB 5000:IF Z<14 THEN PRINT"MAX SPEED AT NEXT BEND IS ";MX
230 FOR DL=1TO A2
240 D=16:GOSUB 5000:PRINT"GEAR=";G;"REVS=";INT(R);"MPH=";INT(S);"[CL
"
250 FORZZ=1TO500:NEXT
260 POKE X,145:POKE Y,42:Y=X:X=X+B2
500 GET N
504 IF N=0 AND NN=0 THEN 550
506 IF N=0 AND NN=1 THEN NN=0:GOTO 545
510 IF N=7 THEN G=G+1:IF G>5 THEN G=4:GOTO 620
520 IF N=4 THEN G=G-1:IF G<1 THEN G=1:GOTO 620
530 IF N=8 OR N=5 OR N=2 THEN GOSUB 2000
540 IF N=9 THEN GOSUB 3000
542 NN=1:GOTO 500
545 GOSUB 4000
550 IF S<2 THEN 605
560 IF R<2000 THEN 600
565 IF R<12000 THEN 590
570 IF R>14000 THEN 610
575 OV=OV+1:IF OV>2 THEN 610
580 D=17:GOSUB 5000:PRINT"[RVS]WARNING[OFF] YOU HAVE OVERREVED":S=S-S/4
590 NL=1:IF S<200 THEN NL=INT((200-S)/8)
595 GOSUB 1000:D=18:GOSUB 5000:GOTO 650
600 D=17:GOSUB 5000:PRINT"[RVS]YOU HAVE STALLED[OFF]":GOTO 608
605 D=17:GOSUB 5000:PRINT"[RVS]YOU HAVE BRAKED TO A STANDSTILL[OFF]
608 B=0:G=1:S=0:A=3:GOSUB 4000:GOTO590
610 D=17:GOSUB 5000:PRINT"[RVS]YOU HAVE BLOWN UP":END
620 D=17:GOSUB 5000:PRINT"[RVS]CHECK YOUR GEARS":GOTO 590
650 NEXT DL
660 X=X+C2
670 IF S<MX OR Z=14 THEN 699
680 IF S>(MX+MX/5) THEN D=17:GOSUB 5000:PRINT"[RVS]YOU HAVE CRASHED[OFF]":END
690 D=17:GOSUB 5000:PRINT"[RVS]YOU HAVE SKIDDED & LOST SPEED[OFF]"
694 S=S-S/3
699 NEXT Z
700 TS=(TI-S9)/60:TM=INT(TS/60)
705 TS=TS-(TM*60)
710 D=17:GOSUB 5000:
720 PRINT"YOUR LAP TIME WAS";TM;"MINS";TS;"SECS
730 END
1000 FOR L=0TONL:L1=SQR(NL):NEXT
1009 RETURN
2000 IF N=8 THEN VV=3
2002 IF N=5 THEN VV=2
2003 IF N=2 THEN VV=1
2005 B=B+VV:A=0
2010 IF B>9 THEN B=9
2030 RETURN
3000 A=A+1:IF A>9 THEN A=9
3010 B=0
3020 RETURN
4000 S=S+(A*3.2*GS(G))-BS(B)-S/20
4010 R=S*GS(G)*61
4020 RETURN
5000 PRINT"[HOME]
5005 IF D>16 THEN D=D-1:ML=1
5010 FOR D1=1 TO D
5014 PRINT
5016 NEXT
5017 IF ML=0 THEN 5020
5018 ML=0:PRINT"
5020 RETURN
6000 END
READY.
```

BUMBLE BEE - VIC 20

```
100 REM PET BENELUX
110 REM EXCHANGE
120 REM HOLLAND
130 HA=36878:VI=36876
140 POKE36879,42:PRINT"■ *** BUMBLE BEE ***■■
150 POKE36869,240
160 PRINT"  HIT ANY KEY TO STOP■■
170 POKEHA,12:POKEV1,0
180 RESTORE
190 FORD=1TO10000
200 READPITCH$
210 GET#1,IFW$<>""THEN1100
220 IFPI$="N"THENNO=10000
230 IFPI$="R"THENXX=0
240 IFPI$="C"THENXX=135
250 IFPI$="DF"THENXX=143
260 IFPI$="D"THENXX=147
270 IFPI$="EF"THENXX=151
280 IFPI$="E"THENXX=159
290 IFPI$="F"THENXX=163
300 IFPI$="GF"THENXX=167
310 IFPI$="G"THENXX=175
320 IFPI$="AF"THENXX=179
330 IFPI$="A"THENXX=183
340 IFPI$="BF"THENXX=187
350 IFPI$="B"THENXX=191
360 IFPI$="H"THENXX=193
370 IFPI$="DF1"THENXX=199
380 IFPI$="D1"THENXX=201
390 IFPI$="EF1"THENXX=203
400 IFPI$="E1"THENXX=207
410 IFPI$="F1"THENXX=209
420 IFPI$="GF1"THENXX=212
430 IFPI$="G1"THENXX=215
440 IFPI$="A1"THENXX=219
450 IFPI$="BF1"THENXX=221
460 IFPI$="B1"THENXX=223
470 IFPI$="AF1"THENXX=217
480 IFPI$="C2"THENXX=225
490 IFPI$="DF2"THENXX=233
500 IFPI$="B2"THENXX=140
510 IFPI$="D2"THENXX=228
520 POKEV1,XX
530 PRINTPI$" ";
540 REM   FOR PP=1 TO 24:NEXT
550 NEXT
560 DATAR,E,EF,D,DF,C,F,E,EF
570 DATAE,EF,D,DF,C,DF,D,EF
580 DATAE,EF,D,DF,C,F,E,EF
590 DATAE,EF,D,DF,C,DF,D,EF
600 DATAE,EF,D,DF,C,DF,C,B3
610 DATAC,DF,D,EF,E,F,E,EF
620 DATAE,EF,D,DF,D,DF,C,B3
630 DATAC,DF,D,EF,E,GF,G,AF
640 DATAA,AF,G,GF,F,BF,A,AF
650 DATAA,AF,G,GF,F,GF,G,AF
660 DATAA,AF,G,GF,F,BF,A,AF
670 DATAA,AF,G,GF,F,GF,G,AF
680 DATAA,AF,G,GF,G,GF,F,E
690 DATAF,GF,G,AF,A,BF,A,AF
700 DATAA,AF,G,GF,G,GF,F,E
710 DATAF,GF,G,AF,A,BF,A,AF
720 DATAA1,A1,A,BF,A,BF,A,BF
730 DATABF1,BF1,A,BF,A,BF,A,BF
740 DATAA1,A1,A,BF,A,BF,A,BF
750 DATABF1,BF1,A,BF,A,BF,A,BF
760 DATAA1,A1,A,BF,BF1,BF1,A,BF
770 DATAA1,BF,BF1,BF,B1,BF,C2,BF
780 DATADF2,DF2,B,H,DF1,H,B,BF,A,BF,B,H,DF1,H,B,BF
790 DATAD1,D1,D,EF,D,EF,D,EF
800 DATAEF1,EF1,D,EF,D,EF,D,EF
810 DATAD1,D1,D,EF,D,EF,D,EF,EF1,EF1,D,EF,D,EF,D,EF
820 DATAD1,D1,D,EF,EF1,EF1,D,EF,D1,EF,EF1,EF,E1,EF,F1,EF
830 DATAGF1,GF1,F,E,F,GF,F,E,EF
840 DATAD,EF,E,F,GF,G,AF,A
850 DATABF,DF1,H,B,BF,EF1,D1,DF1
860 DATAD1,DF1,H,B,BF,B,H,DF1
870 DATAD1,DF1,H,B,H,B,BF,A
880 DATABF,B,H,DF1,H,DF1,D1,EF1
890 DATAE1,EF1,D1,DF1,D1,DF1,H,B
900 DATAH,B,BF,A,AF,G,GF,F
910 DATAE,E,R,R,E1,R,H,R
920 DATAA,R,F,R,A,R,H,R
930 DATAE1,R,R,R,E1,R,H,R
940 DATAA,R,F,R,A,R,H,R
950 DATAE1,R,R,R,R,R,R,R
960 DATAC,DF,D,EF,E,F,GF,G,AF,A,BF,B,H,DF1,D1,EF1
970 DATAE1,F1,E1,EF1,E1,F1,E1,EF1,E1,EF1,D1,DF1,H,F1,E1,EF1
980 DATAE1,EF1,D1,DF1,H,DF1,D1,EF1,E1,EF1,D1,DF1,D1,EF1
990 DATAE1,F1,E1,EF1,E1,F1,E1,EF1,E1,EF1,D1,DF1,H,F1,E1,EF1
1000 DATAE1,EF1,D1,DF1,H,DF1,D1,EF1,E1,EF1,D1,DF1,D1,EF1
1010 DATAE1,EF1,D1,DF1,D1,DF1,H,B,H,DF1,D1,EF1,E1,F1,E1,EF1
1020 DATAE1,EF1,D1,DF1,D1,DF1,H,B,H,DF1,D1,EF1,E1,GF1,G1,AF1
1030 DATAA1,AF1,G1,GF1,G1,GF1,F1,E1,F1,E1,EF1,D1,DF1,H,B,BF
1040 DATAA,A,R,R,E,F,GF,G,AF,A,BF,B,H,DF1,D1,EF1
1050 DATAE,F,GF,G,AF,A,BF,B,H,DF1,D1,EF1
1060 DATAE1,GF1,G1,AF1,A1,R,R,R,R,R,H
1070 DATAA1,R,R,R,R,R,R,R,A,A,A,A,R,R,R,R
1080 DATAN
1090 GOTO180
1100 POKEHA,0:POKEV1,0:POKE36879,27:PRINT"■":END
READY.
```

VIC FOR THE BIRDS

```
1 VIC=9*16↑3:POKE VIC,11:PRINT"■":PRINT
2 PRINT"■ DEMONSTRATION OF"
3 PRINT" USER-DEFINED"
4 PRINT" CHARACTERS AND REAL-"
5 PRINT" TIME ANIMATION. "
6 PRINT" ADOPTED BY PBE-NL  ■■■■■■■■■■■■■BY BOB HAINES"
7 FOR I=1TO6000:NEXT I
9 VM=7680
10 CNVB=30*30
12 CM=VM-512
20 FOR A=0 TO 506
30 POKE VM+A,0
40 POKE CNVB+A,1
50 NEXT A
60 FOR A=0 TO 7
70 POKE CM+A,0
71 NEXT A
72 FOR A=0 TO 7
73 POKE CM+56+A,16
74 NEXT A
75 POKE VIC+5,255
80 POKE CM+62,56
81 POKE CM+48,24
82 POKE CM+49,153
83 POKE CM+50,90
84 POKE CM+51,60
85 POKE CM+52,90
86 POKE CM+53,189
87 POKE CM+54,60
88 POKE CM+55,126
89 FORTT=1TO5
90 POKE CNVB+30,7
91 POKE CNVB+31,7
92 POKE CNVB+32,7
93 POKE CNVB+67,1
94 POKE CNVB+68,1
95 POKE CNVB+69,1
96 POKE CNVB+205,6
97 POKE CNVB+206,6
98 POKE CNVB+207,6
99 POKE VM+30,1
100 POKE VM+31,2
101 POKE VM+32,3
102 POKE VM+67,1
103 POKE VM+68,2
104 POKE VM+69,3
105 POKE VM+205,1
106 POKE VM+206,2
107 POKE VM+207,3
120 POKE CM+16,126
130 POKE CM+17,60
140 POKE CM+18,153
150 POKE CM+19,255
160 POKE CM+20,255
170 POKE CM+21,126
180 POKE CM+22,24
190 POKE CM+23,24
192 POKE VIC+15,239
194 PRINT"■■■■■■■■■■■■■■■■■■■■";
200 PRINT"■ACACACACACACACACACAC";
210 PRINT"■■@ACACA@@@@CACA@@CAC@AC";
211 PRINT"■■@@AC@@@CACACA@@ACAC@@@";
212 PRINT"■■@@@ACACAC@@ACAC@CAC@@@";
213 PRINT"■CAC@@CAC@@ACACACA@@AC@ACAC@@";
214 PRINT"■■@@ACACA@@CAC@@ACA@@@AC";
220 PRINT"■■■■■■■■■■■■ABC■■■■■■■■■■ABC■■■■■■■ABC"
230 PRINT"■■■■■■■■■■■■■■■■■EDDDDDD■■■■■■■EDDDDD■■■■■■EDDDD■■■EDD■■E"
234 FOR A=0 TO 9
235 POKE CM+32+A,255
236 NEXT A
237 POKE CM+42,127
238 POKE CM+43,127
239 POKE CM+44,63
240 POKE CM+45,31
241 POKE CM+46,15
242 POKE CM+47,3
247 FOR Z=1 TO 5
248 RESTORE
250 FOR A=1 TO 7
260 FOR B=0 TO 7
270 READ C
280 POKE CM+8+B,C
290 READ C
300 POKE CM+24+B,C
310 NEXT B
320 NEXT A
331 FOR A=1 TO 7
```

```
332 FOR B=0 TO 7
333 READ C
334 POKE CM+8+B,C
335 READ C
336 POKE CM+24+B,C
337 NEXT B
338 NEXT A
340 NEXT Z
365 FOR A=1 TO 12
370 READB:READC
380 POKE VIC+12,B
385 POKE VIC+14,15
390 FOR X=1TO25*C:NEXTX
400 FOR Y=14 TO 0 STEP-1
401 POKE VIC+14,Y
403 NEXT Y
405 READ C
406 FOR X=1 TO 25*C:NEXTX
411 NEXT A
412 POKE VIC+12,0
500 PRINT"■■■■■■■■■■■■■■■■■■■■■▮▮▮▮▮▮▮▮▮F";
501 FOR X= 1 TO 50:NEXT:PRINT"▮▮A▮▮";
502 POKE VIC+14,15
510 FOR Y=1 TO 7
520 GOSUB 599
530 NEXT
540 PRINT"▮▮▮@F@";
541 FOR X=1 TO 20:NEXTX:PRINT"▮▮▮▮@@"
542 POKE VIC+13,170
543 FOR Y=15 TO 0 STEP-1:POKEVIC+14,Y:FOR X=1 TO 30:NEXT X:NEXT Y
544 POKE VIC+13,0
545 FOR X=1 TO 20:NEXTX
547 FOR X=1 TO700:NEXT
590 NEXTTT:FOR I=1TO6000:NEXTI:POKE36869,240:RUN
599 PRINT"▮▮▮▮▮G";
600 POKE VIC+13,240+Y
601 FOR X=1 TO 45:NEXT:RETURN
1000 DATA0,0,0,0,0,0,0,3,192,15,240,112,14,128,1
1010 DATA0,0,0,0,0,1,128,3,192,60,60,192,3,0,0
1020 DATA0,0,0,0,0,1,128,15,240,240,15,0,0,0,0
1030 DATA0,0,0,1,128,255,255,1,128,0,0,0,0,0
1040 DATA0,0,0,240,15,15,240,1,128,0,0,0,0,0
1050 DATA0,0,192,3,60,60,3,192,1,128,0,0,0,0,0
1060 DATA128,1,112,14,15,240,3,192,0,0,0,0,0,0,0
1070 DATA128,1,112,14,15,240,3,192,0,0,0,0,0,0,0
1080 DATA0,0,192,3,60,60,3,192,1,128,0,0,0,0,0
1090 DATA0,0,0,240,15,15,240,1,128,0,0,0,0,0
1100 DATA0,0,0,1,128,255,255,1,128,0,0,0,0,0
1110 DATA0,0,0,0,0,1,128,15,240,240,15,0,0,0,0
1120 DATA0,0,0,0,0,1,128,3,192,60,60,192,3,0,0
1130 DATA0,0,0,0,0,0,0,3,192,15,240,112,14,128,1
2000 DATA 213,8,2,217,8,2,204,3,1,198,1,1,191,2,1
2010 DATA179,8,2,170,8,2,179,4,1,191,1,1,198,1,1,213,3,1,217,15,0
READY.
```

SPACEWAR PART 1 : VIC20

```
100 POKE36879,27:PRINT"☐      VIC SPACEWAR"
110 PRINT"■■■■JUST A MOMENT...."
140 X=PEEK(56)-2:POKE56,X:POKE52,X:POKE51,PEEK(55):CLR
150 CS=256*PEEK(52)+PEEK(51):S=7680
160 FORI=CSTOCS+511:POKEI,PEEK(I+32768-CS):NEXT
170 READX:IFX<0THEN200
180 FORI=XTOX+7:READJ:POKEI,J:NEXT
190 GOTO170
200 INPUT"■■   INSTRUCTIONS N▮▮▮▮";A$
210 IFA$="N"THEN2000
220 PRINT"☐YOU ARE ENGAGED IN A"
230 PRINT"BATTLE TO THE DEATH"
240 PRINT"WITH A KILL-CRAZED,"
250 PRINT"KAMIKAZE MINGON IN"
260 PRINT"THE ASTEROID BELT OF"
270 PRINT"MONGO."
280 PRINT"■YOUR SHIP IS PURPLE"
290 PRINT"AND THE MINGON SHIP"
300 PRINT"IS DARK BLUE."
310 PRINT"■THE JOYSTICK CONTROLS"
320 PRINT"YOUR DIRECTION. THE"
330 PRINT"RED BUTTON FIRES YOUR"
340 PRINT"LASER MISSILES. MAKE"
350 PRINT"A HYPERSPACE JUMP BY"
360 PRINT"PRESSING THE SPACE"
380 PRINT"BAR."
390 PRINT"■PRESS ANY KEY"
400 A$="":GETA$:IFA$=""THEN400
410 PRINT"☐YOU LOSE THE BATTLE"
420 PRINT"IF YOU:"
430 PRINT"■1 ARE BLASTE BY A"
440 PRINT"   MINGON'S LASER"
450 PRINT"■2 ARE SUCKED INTO A"
460 PRINT"   BLACK HOLE"
470 PRINT"■3 COLLIDE WITH AN"
480 PRINT"   ASTEROID"
490 PRINT"■4 ARE PULLED INTO"
500 PRINT"   THE SUN BY GRAVITY"
510 PRINT"■5 ARE RAMMED BY THE"
520 PRINT"   THE MINGON SHIP"
530 PRINT"■PRESS ANY KEY"
540 A$="":GETA$:IFA$=""THEN540
2000 PRINT"☐   ENTER SKILL LEVEL"
2010 PRINT"■■     ■1■-TRIVIAL"
2020 PRINT"■        TO"
2030 PRINT"■     ■9■-IMPOSSIBLE"
2040 A$="":GETA$:IFA$=""THEN2040
2050 SL=VAL(A$):IFSL<1ORSL>9THEN2000
2060 PRINT"☐  SPACEWAR OPTIONS"
2110 PRINT"■■     ■S■UN'S GRAVITY"
2120 PRINT"■      ■B■LACK HOLE"
2130 PRINT"■      ■N■O GRAVITY"
2140 PRINT"■■  SELECT ONE OPTION":G=0
2150 A$="":GETA$:IFA$=""THEN2150
2160 IFA$="S"THENG=1
2170 IFA$="B"THENG=2
2180 POKES-2,SL:POKES-1,G
2185 PRINT"■■JUST A FEW MOMENTS■"
2190 POKE198,5:POKE631,78:POKE632,69:POKE633,87:POKE634,13:POKE635,131:END
9800 DATA7168,192,240,127,103,34,54,63,50
9810 DATA7176,4,14,62,227,227,62,14,4
9820 DATA7184,50,63,54,34,103,127,240,192
9830 DATA7192,24,24,60,36,102,231,126,24
9840 DATA7208,24,126,231,102,36,60,24,24
9850 DATA7216,3,15,254,230,68,108,252,76
9860 DATA7224,32,112,124,199,199,124,112,32
9870 DATA7232,76,252,108,68,230,254,15,3
9871 DATA7240,0,0,12,28,56,48,0,0
9872 DATA7248,0,0,48,56,28,12,0,0
9873 DATA7256,0,24,24,24,24,24,0,0
9874 DATA7264,0,0,0,62,62,0,0,0
9900 DATA7296,0,112,126,102,32,48,48,0
9910 DATA7304,228,18,37,68,36,18,33,198
9920 DATA7312,0,48,48,32,102,126,112,0
9930 DATA7320,68,170,145,0,34,85,137,129
9940 DATA7328,153,90,60,255,255,60,90,153
9950 DATA7336,129,137,85,34,0,145,170,68
9960 DATA7344,0,14,126,102,4,12,12,0
9970 DATA7352,198,33,18,36,68,37,18,228
9980 DATA7360,0,12,12,4,102,126,14,0,-1
READY.
```

SPACEWAR PART 2 : VIC20

```
1 C=22:R=23:S=7680:A=30720:DD=37154:P1=37151:P2=37152:SL=PEEK(S-2)
2 POKE36879,27:T=3:CY=4:CE=6:CA=0:CS=7:CQ=0:CM=0:X=-1:G=PEEK(S-1)
3 V=36878:S1=V-2:S2=V-1:DIMDC%(2,2),OC%(2,2)
4 FORI=0TO2:FORJ=0TO2:X=X+1:DC%(I,J)=X:OC%(I,J)=X+16:NEXTJ,I
5 DEFFNA(Z)=S+X+C*Y:DEFFNB(Z)=PEEK(FNA(Z)):DEFFNR(Z)=INT(RND(1)*Z)
6 POKEV-9,255:PRINT"☐■":X=S+A:FORI=XTOX+505:POKEX,T:NEXT
7 POKE36879,62:FORI=1TO3+2*SL:X=FNR(C):Y=FNR(R)
8 POKEFNA(0),46:POKEFNA(0)+A,CA:NEXT
9 IFG=1THENX=11:Y=12:POKEFNA(0),20:POKEFNA(0)+A,CS:SX=X:SY=Y
10 IFG=2THENSX=FNR(C):SY=FNR(R)
11 X=FNR(C):Y=FNR(R):IFFNB(0)<>32THEN11
12 D=X:E=Y:Q=FNA(0):U=-1:O=FNR(3)-1
13 POKEFNA(0),DC%(U+1,O+1):POKEFNA(0)+A,CY
14 X=FNR(C):Y=FNR(R):IFFNB(0)<>32THEN14
15 H=X:L=Y:K=FNA(0):M=1:N=FNR(3)-1
16 POKEFNA(0),DC%(M+1,N+1):POKEFNA(0)+A,CE
17 A$="":GETA$:IFA$=" "THEND=FNR(C):E=FNR(R)
18 GOSUB75:IFFBTHENX=D:Y=E:PX=U:PY=O:GOSUB47
19 B=0:F=0:IFJ0THENB=1
20 IFJ2THENB=-1
21 IFJ1THENF=1
22 IFJ3THENF=-1
23 IFB=0ANDF=0THENB=U:F=O
24 U=B:O=F:IFG=0THEN28
25 B=SX-D:F=SY-E:J=SQR(B*B+F*F):J=(20-J)/30:J=1-J*J
26 IFRND(1)<JTHEN28
27 B=SGN(B):F=SGN(F):D=D+B:E=E+F:GOTO29
28 D=D+U:E=E+O
29 IFE<0THENE=R
30 IFE>RTHENE=0
31 IFD>CTHEND=0
32 IFD<0THEND=C
33 X=D:Y=E:J=FNB(0):IFJ=32THEN35
34 IFJ=46ORJ=20ORFNA(0)=KTHEN60
35 IFX=SXANDY=SYTHENA$="WERE SUCKED INTO    A BLACK HOLE!":WC=WC+1:GOTO71
36 POKEQ,32:POKEQ+A,T:Q=FNA(0):POKEQ,DC%(U+1,O+1):POKEQ+A,CY
37 J=0:IFFNR(9)>SLTHEN39
38 M=D-H:N=E-L:M=SGN(M):N=SGN(N):IFM=0ORN=0THENJ=1
39 H=H+M:L=L+N:IFH<0THENH=C
40 IFH>CTHENH=0
41 IFL>RTHENL=0
42 IFL<0THENL=R
43 X=H:Y=L:IFFNB(0)<>32THENM=FNR(3)-1:N=FNR(3)-1:GOTO39
44 POKEK,32:POKEK+A,T:K=FNA(0):POKEK,DC%(M+1,N+1):POKEK+A,CE
45 IFRND(1)<.1OR(J=1ANDFNR(9)<SL)THENPX=M:PY=N:GOSUB47
46 GOTO17
47 Z=PX*PY:POKEV,8:IFZ=1THENJ=10
48 IFZ=-1THENJ=9
49 IFZ=0ANDPX=0THENJ=11
50 IFZ=0ANDPY=0THENJ=12
51 FORI=1TO10:X=X+PX:Y=Y+PY:POKES2,230-I
52 IFI<>1THENPOKEZ,32:POKEZ+A,T:IFX>CTHENX=0
53 IFX<0THENX=C
54 IFY>RTHENY=0
55 IFY<0THENY=R
56 B=FNB(0):IFB=32THEN58
57 IFB=46ORB=20ORFNA(0)=KORFNA(0)=QTHENI=10:NEXTI:GOTO60
58 Z=FNA(0):POKEZ,J:POKEZ+A,CM:NEXTI
59 POKEZ,32:POKEZ+A,T:POKEV,0:RETURN
60 POKES2,230:SC=X-1:IFSC<0THENSC=0
61 FC=X+1:IFFC>CTHENFC=C
62 SR=Y-1:IFSR<0THENSR=0
63 FR=Y+1:IFFR>RTHENFR=R
64 FORX=SCTOFC:FORY=SRTOFR:J=OC%(X-SC,Y-SR)
65 POKEFNA(0),J:POKEFNA(0)+A,CQ:NEXTY,X
66 POKES1,220:FORJ=15TO0STEP-1:POKEV,J:FORJ1=1TO50:NEXTJ1,J
67 POKEV,0:FORX=SCTOFC:FORY=SRTOFR:POKEFNA(0),32:POKEFNA(0)+A,T:NEXTY,X
68 IFPEEK(Q)=32THENA$="WERE VAPORIZED!":WC=WC+1:GOTO71
69 IFPEEK(K)=32THENA$="TRIUMPHED!":WH=WH+1:GOTO71
70 RETURN
71 POKEV-9,240:PRINT"☐■■■YOU  ";A$
72 PRINT"■■SCORE: VIC";WC;"  YOU";WH
73 FORJ=1TO500:NEXT:GOTO78
74 END
75 POKEDD,127:P=PEEK(P2)AND128:J0=-(P=0):POKEDD,255
76 P=PEEK(P1):J1=-((PAND8)=0):J2=-((PAND16)=0)
77 J3=-((PAND4)=0):FB=-((PAND32)=0):RETURN
78 IFWH=5THEN81
79 IFWC=5THEN83
80 GOTO6
81 PRINT"☐   CONGRATULATIONS!"
82 GOTO84
83 PRINT"☐TO BAD"
84 FORJ=1TO1000:NEXT:RUN
READY.
```

BLACKBOX

```
10 REM*****BLACK BOX*****
20 REM**              **
30 REM***PROGRAMMED BY***
40 REM**              **
50 REM*****R.C.MERRY*****
60 REM**              **
70 REM******************
80 REM******************
90 REM***INSTRUCTIONS***
100 PRINT"[CLR]            [RVS]BLACK[OFF] [RVS]BOX[OFF]"
110 PRINT"[4CD]THIS IS THE GAME OF BLACK BOX, THE GAME"
120 PRINT"[CD]OF DEDUCTION BY WADDINGTONS. YOU WILL"
130 PRINT"[CD]NEED THE STANDARD RULES OF BLACK BOX"
140 PRINT"[CD]TO PLAY. THE COMPUTER GENERATES RANDOM"
150 PRINT"[CD]PATTERNS OF ATOMS. HOW MANY ATOMS"
160 INPUT"[CD]WOULD YOU LIKE IN THE FIRST GAME";Z
170 REM***INITIALIZE***
180 A$="[39CR]"
190 D$="[24CD]"
200 L$="  |":M$="——+":RT=0:GT=0
210 B$="ABCDEFGHIJKLMNOP":DIMP(9,9)
220 S=0:B=0
230 REM**CLEAR ARRAY,LOAD RAY NUMBERS**
240 FORX=0TO9:FORY=0TO9:P(X,Y)=0:NEXTY,X
250 FORY=1TO8:P(0,Y)=Y:P(9,Y)=25-Y:NEXT
260 FORX=1TO8:P(X,0)=33-X:P(X,9)=X+8:NEXT
270 REM***PLACE RANDOM ATOMS***
280 FORI=1TOZ
290 RX(I)=INT(8*RND(1)+1):RY(I)=INT(8*RND(1)+1)
300 IFP(RX(I),RY(I))<>0THEN290
310 P(RX(I),RY(I))=99:NEXT
320 REM*PRINT DISPLAY DURING BLANKING*
330 POKE59409,52
340 PRINT"[CLR]";
350 FORI=1TO9
360 FORJ=1TO9:PRINTL$;:NEXT:PRINT
370 FORJ=1TO9:PRINTM$;:NEXT:PRINT"——":NEXT
380 FORJ=1TO9:PRINTL$;:NEXT
390 PRINT"[HOME,CR,2CD]1[CL,2CD]2[CL,2CD]3[CL,2CD]4[CL,2CD]5[CL,2CD]";
395 PRINT"6[CL,2CD]7[CL,2CD]8[2CD,2CR]9[CR]10[CR]11[CR]12[CR]13[CR]14[CR]15[CR]
16"
400 PRINT"[HOME,3CR]32[CR]31[CR]30[CR]29[CR]28[CR]27[CR]26[CR]25"
410 PRINT"[HOME]";LEFT$(A$,27);"[2CD]24[2CD,2CL]23[2CD,2CL]22[2CD,2CL]";
415 PRINT"21[2CD,2CL]20[2CD,2CL]19[2CD,2CL]18[2CD,2CL]17[2CD]"
420 REM USE WITH CAUTION:POKE59409,60
430 REM***INPUT OPTION***
440 PRINT"[HOME]";LEFT$(D$,20);"DO YOU WANT TO 1)INPUT A RAY"
450 INPUT"                   2)SEE THE ANSWER";R
460 IFR=1ORR=2THEN480
470 PRINT"PLEASE ENTER 1 OR 2";:GOTO440
480 GOSUB1350
490 ONRGOTO500,1250
500 REM***INPUT RAY NUMBER***
510 PRINT"[HOME]";LEFT$(D$,20);:INPUT"RAY NUMBER";G
```

Basic Programs

```
520 IFG<10RGD>32THENPRINT"NOT A VALID RAY":GOSUB1350:GOTO510
530 N=INT((G-1)/8)+1
540 ONNGOTO560,660,760 ,860
550 REM***LEFT TO RIGHT***
560 X=0:Y=G
570 IFP(X+1,Y)=99THEN960
580 IFP(X+1,Y-1)=99ANDP(X+1,Y+1)=99THEN1050
590 IFP(X+1,Y-1)=99ANDX=0THEN1050
600 IFP(X+1,Y+1)=99ANDX=0THEN1050
610 IFP(X+1,Y-1)=99THEN870
620 IFP(X+1,Y+1)=99THEN670
630 X=X+1:IFX<>9THEN570
640 GOTO1120
650 REM***BOTTOM TO TOP***
660 X=G-8:Y=9
670 IFP(X,Y-1)=99THEN960
680 IFP(X+1,Y-1)=99ANDP(X+1,Y-1)=99THEN1050
690 IFP(X-1,Y-1)=99ANDY=9THEN1050
700 IFP(X+1,Y-1)=99ANDY=9THEN1050
710 IFP(X-1,Y-1)=99THEN570
720 IFP(X+1,Y-1)=99THEN770
730 Y=Y-1:IFY<0THEN670
740 GOTO1120
750 REM***RIGHT TO LEFT***
760 X=9:Y=25-G
770 IFP(X-1,Y)=99THEN960
780 IFP(X-1,Y-1)=99ANDP(X-1,Y+1)=99THEN1050
790 IFP(X-1,Y-1)=99ANDX=9THEN1050
800 IFP(X-1,Y+1)=99ANDX=9THEN1050
810 IFP(X-1,Y-1)=99THEN870
820 IFP(X-1,Y+1)=99THEN670
830 X=X-1:IFX<0THEN770
840 GOTO1120
850 REM***TOP TO BOTTOM***
860 X=33-G:Y=0
870 IFP(X,Y+1)=99THEN960
880 IFP(X-1,Y+1)=99ANDP(X+1,Y+1)=99THEN1050
890 IFP(X-1,Y+1)=99ANDY=0THEN1050
900 IFP(X+1,Y+1)=99ANDY=0THEN1050
910 IFP(X-1,Y+1)=99THEN570
920 IFP(X+1,Y+1)=99THEN770
930 Y=Y+1:IFY>9THEN870
940 GOTO1120
950 REM***ABSORBED RAY***
960 S=S+1:GOSUB1380:PRINT"[HOME]";LEFT$(D$,20);"RAY WAS ABSORBED"
970 ONNGOTO980 ,990 ,1000,1010
980 PRINT"[HOME]";LEFT$(D$,2*G);" [RVS]◆[OFF]":GOTO1020
990 PRINT"[HOME]";LEFT$(D$,18);LEFT$(A$,3*(G-8));" [RVS]◆[OFF]":GOTO1020
1000 PRINT"[HOME]";LEFT$(D$,2*(25-G));LEFT$(A$,27);" [RVS]◆[OFF]":GOTO1020
1010 PRINT"[HOME]";LEFT$(A$,3*(33-G));" [RVS]◆[OFF]"
1020 FORI=1TO3000:NEXT
1030 GOSUB1350:GOTO430
1040 REM***REFLECTED RAY***
1050 S=S+1:GOSUB1380:PRINT"[HOME]";LEFT$(D$,20);"RAY WAS REFLECTED"
1060 ONNGOTO1070,1080,1090,1100
1070 PRINT"[HOME]";LEFT$(D$,2*G);" [RVS]*[OFF]":GOTO1110
1080 PRINT"[HOME]";LEFT$(D$,18);LEFT$(A$,3*(G-8));" [RVS]*[OFF]":GOTO1110
1090 PRINT"[HOME]";LEFT$(D$,2*(25-G));LEFT$(A$,27);" [RVS]*[OFF]":GOTO1110
1100 PRINT"[HOME]";LEFT$(A$,3*(33-G));" [RVS]*[OFF]"
1110 FORI=1TO3000:NEXT:GOSUB1350:GOTO430
1120 REM***RAY EMERGES***
1130 S=S+2:B=B+1:GOSUB1380:PRINT"[HOME]";LEFT$(D$,20);"RAY EMERGED AT ";P(X,Y)
1140 IFX=0THEN1170
1150 IFY=0THEN1180
1160 PRINT"[HOME]";LEFT$(A$,3*X);LEFT$(D$,2*Y);" ";MID$(B$,B,1):GOTO1190
1170 PRINT"[HOME]";LEFT$(D$,2*Y);" ";MID$(B$,B,1):GOTO1190
1180 PRINT"[HOME]";LEFT$(A$,3*X);" ";MID$(B$,B,1)
1190 ONNGOTO1200,1210,1220,1230
1200 PRINT"[HOME]";LEFT$(D$,2*G);" ";MID$(B$,B,1):GOTO1240
1210 PRINT"[HOME]";LEFT$(D$,18);LEFT$(A$,3*(G-8));" ";MID$(B$,B,1):GOTO1240
1220 PRINT"[HOME]";LEFT$(D$,2*(25-G));LEFT$(A$,27);" ";MID$(B$,B,1):GOTO1240
1230 PRINT"[HOME]";LEFT$(A$,3*(33-G));" ";MID$(B$,B,1)
1240 FORI=1TO3000:NEXT:GOSUB1350:GOTO430
1250 REM***PRINT ANSWER,FINAL SCORE***
1256 FORI=1TO2:PRINT"[HOME]";LEFT$(A$,RX(I)*3);LEFT$(D$,RY(I)*2);"●":NEXT
1270 PRINT"[HOME]";LEFT$(D$,20);"HOW MANY DID YOU GET WRONG";
1280 INPUTW
1290 S=S+5*W:GOSUB1350
1300 PRINT"[HOME]";LEFT$(D$,20);"YOUR FINAL SCORE WAS ";S:RT=RT+S:GT=GT+1
1310 INPUT"[2CD]ANOTHER GAME (Y/N)";R$
1320 IFLEFT$(R$,1)="N"THEN1390
1330 INPUT"[CD]HOW MANY ATOMS";Z:GOTO220
1340 REM***CLEAR BOTTOM OF SCREEN***
1350 PRINT"[HOME]";LEFT$(D$,19);
1360 FORI=1TO200:PRINT" ":NEXT:RETURN
1370 REM***PRINT SCORE***
1380 PRINT"[HOME]";LEFT$(A$,30);LEFT$(D$,5);"[RVS]SCORE[OFF]:";S:RETURN
1390 REM***WORK OUT FINAL AVERAGE***
1400 AV=RT/GT
1410 PRINT"[CLR]YOUR MEAN SCORE WAS:";AV
1420 PRINT"[CD]YOU PLAYED:";GT;"GAMES"
1430 IFAV>20THENG$="POOR!!"
1440 IFAV<=20THENG$="FAIR"
1450 IFAV<15THENG$="ABOUT AVERAGE"
1460 IFAV<13THENG$="GOOD"
1470 IFAV<10THENG$="EXCELLENT!!"
1480 PRINT"[CD]YOUR PERFORMANCE WAS ";G$
1490 END
READY.
```

DISK CHECKER : JIM BUTTERFIELD

```
100 PRINT"DISK CHECKER  - JIM BUTTERFIELD"
110 DIM A(255),CX(77,28),DX(1),N$(224),TX(224,1),SX(224,1),LX(224),RX(77)
120 DX(0)=58:DX(1)=42:Z$=CHR$(0)
130 DATA 1,17,20,24,19,30,17,35,16,0
140 DATA 65,17,20,24,18,30,17,35,16,0
150 DATA 67,39,28,53,26,64,24,77,22,0
160 B$=CHR$(17):INPUT"DRIVE#";D$:IFD$="S"THEND$="0":B$=CHR$(3)
170 IFD$<>"0"ANDD$<>"1"GOTO160
180 OPEN15,8,15,"I"+D$:GOSUB3020
190 OPEN3,8,3,"#"+D$:GOSUB3020
200 A0=1:GET#3,A$:A=ASC(A$+Z$)
210 READA1:IFA=A1GOTO250
220 F%=F%+1:IFF%=3GOTO290
230 READA1:IFA1=0GOTO210
240 GOTO230
250 READA1:IFA1=0GOTO270
260 READB1:FORJ=A0TOA1:RX(J)=B1:NEXTJ:A0=J:GOTO250
270 IFA=10RA=65THEND1=1:T9=35:S9=3:D9=18
280 IFA=67THEND1=257:T9=77:S9=4:D9=39
290 IFT9=0THENCLOSE3:PRINT"?? DISK NOT RECOGNIZED ??":STOP
300 REM    GET AND PRINT BAM
310 PRINT"       FREE BLOCK MAP"
320 FORJ=1TOD1:GET#3,A$:NEXTJ
330 FORJ=1TOT9:T1=0
340 IFJ=51THENGET#3,A$,A$,A$,A$,A$
350 GET#3,A$:C=ASC(A$+Z$)
360 PRINTRIGHT$(" "+STR$(J),2);" ";
370 K1=0:FORK=0TOS9-1:GET#3,A$:A=ASC(A$+Z$)
380 FORL=0TO7:AX=A/2:D1=A-AX*2:IFK1<=RX(J)THENCX(J,K1)=D1:PRINTCHR$(DX(D1));
390 A=AX:T1=T1+D1:K1=K1+1:NEXTL,K:PRINT
400 IFT1<>CTHENPRINT"?";
410 NEXTJ
420 REM    DO SPECIFIC JOB
430 PRINT:CLOSE3:PRINT "    CHOOSE --"
440 PRINT "1. CHECK ALL FILES"
450 PRINT "2. CHECK FOR BAD SPOTS"
460 PRINT "3. RECOVER SCRATCHED FILE"
470 PRINT "  YOUR CHOICE? ";
480 GETX$:IFX$=""GOTO480
490 X=ASC(X$)-48:IFX<1ORX>3GOTO480
500 PRINTX$:OPEN2,8,2,"#0":GOSUB3020
510 ONXGOTO520,790,890
520 REM     CHECK FILES
530 T=D9:S=1
540 GOSUB2000
550 FORD=2TO255STEP32:IFA(D)<128GOTO590
560 D3=D3+1:TX(D3,0)=A(D+1):SX(D3,0)=A(D+2):LX(D3)=A(D+28)+A(D+29)*256
570 IFA(D)=132THENTX(D3,1)=A(D+19):SX(D3,1)=A(D+20)
580 N$="":FORK=D+3TOD+18:N$=N$+CHR$(A(K)):NEXTK:N$(D3)=N$
590 NEXTD
600 T=A(0):S=A(1):IFT=D9GOTO540
610 FORD=1TOD3:LX=0
620 PRINTN$(D);
630 T=TX(D,0):S=SX(D,0)
640 IFT>T9ORS<0THENT=0
650 IFT<1ORS>RX(T)THENPRINT" BAD CHAIN":GOTO770
660 IFCX(T,S)=1THENPRINT" UNALLOCATED BLOCKS":GOTO770
670 IFCX(T,S)>1THENPRINT" CONFLICT ";N$(CX(T,S)-1):GOTO770
680 CX(T,S)=1+D
690 GOSUB3000
700 LX=LX+1
710 FORJ=0TO1:PRINT#15,"M-R";CHR$(J);B$:GET#15,A$
720 A(J)=ASC(A$+Z$):NEXTJ
730 T4=T:S4=S:T=A(0):S=A(1):IFT<>0ANDE=0GOTO640
740 T=TX(D,1):S=SX(D,1):TX(D,1)=0:IFT<>0GOTO640
750 IFLX<>LX(D)THENPRINT" INCORRECT BLOCK COUNT":GOTO770
760 PRINT:PRINT"[CU,23SPC,CU]"
770 NEXTD
780 PRINT:PRINTD3;"FILES":GOTO1270
790 REM   SCAN SECTORS
800 IFF%=0THENPRINT"SORRY .. CAN'T DO IT":GOTO1270
810 FORT=1TOT9:PRINT"TRACK";T
820 FORS=0TORX(T)
830 PRINT"[CU,10CR] SECT";S
840 GOSUB3000
850 NEXTS
860 PRINT"[CU,20SPC,CU]"
870 NEXTT
880 PRINT"DISK OK":GOTO1270
890 REM    UNSCRATCH
900 K=0:PRINT"I WILL LOOK FOR DISCARDED FILES..."
910 T=D9:S=1
920 GOSUB2000
930 FORD=2TO255STEP32:IFA(D)<>0ORA(D+1)=0GOTO980
940 IFK=0THENPRINT"DO YOU WANT TO RECOVER:"
950 GETX$:FORK=D+3TOD+18:PRINTCHR$(A(K));:NEXTK:PRINT"? ";
960 GETX$:IFX$<>"Y"ANDX$<>"N"GOTO960
970 PRINTX$:IFX$="Y"GOTO1010
980 NEXTD
990 T=A(0):S=A(1):IFT=D9GOTO920
1000 PRINT"THAT'S ALL ":GOTO1270
1010 T6=T:S6=S:D6=D:T=A(D+1):S=A(D+2):LX(0)=A(D+28)+A(D+29)*256:LX=0
1020 GETX$:PRINT"IS THIS FILE:"
1030 PRINT" 1. SEQUENTIAL"
1040 PRINT" 2. PROGRAM"
1050 PRINT" 3. USR"
1060 IFA(D+19)=0GOTO1080
1070 PRINT" 4. RELATIVE"
1080 PRINT"  WHICH NUMBER? ";
1090 GETX$:IFX$=""GOTO1090
1100 X=ASC(X$)-48:IFX<1ORX>4GOTO1090
1110 PRINTX$:X=X+128
1120 IFX=132THENTX(0,1)=A(D+19):SX(0,1)=A(D+20):IFTX(0,1)=0GOTO1020
1130 IFT>T9ORS<0THENT=0
1140 IFT<1ORS>RX(T)THENPRINT" BAD CHAIN!":GOTO1260
1150 IFCX(T,S)=0THENPRINT" ALLOCATED BLOCKS!":GOTO1260
1160 GOSUB3000:LX=LX+1
1170 FORJ=0TO1:PRINT#15,"M-R";CHR$(J);B$:GET#15,A$
1180 A(J)=ASC(A$+Z$):NEXTJ
1190 T4=T:S4=S:T=A(0):S=A(1):IFT<>0GOTO1130
1200 T=TX(0,1):S=SX(0,1):TX(0,1)=0:IFT<>0GOTO1130
1210 IFLX<>LX(0)THENPRINT" INCORRECT BLOCK COUNT!":GOTO1260
1220 T=T6:S=S6:D=D6
1230 GOSUB3000
1240 PRINT#15,"M-W";CHR$(D);B$;CHR$(1);CHR$(X)
1250 PRINT#15,"U2:2,";D$;T;S:GOSUB3020:GOTO1300
1260 PRINT "SORRY - IT WON'T WORK"
1270 CLOSE2
1280 INPUT"** GOT TIME TO VERIFY/COLLECT DISK";X$
1290 IFASC(X$)=78THENEND
1300 CLOSE2:PRINT#15,"V";D$:END
2000 REM    GRAB FULL DISK BLOCK
2010 GOSUB3000
2020 FORJ=0TO255:PRINT#15,"M-R";CHR$(J);B$:GET#15,A$
2030 A(J)=ASC(A$+Z$):NEXTJ:RETURN
3000 REM    READ BLOCK
3010 PRINT#15,"B-R:2,";D$;T;S
3020 REM   GET ERROR STATUS
3030 INPUT#15,E,E$,E1,E2
3040 IFE<>0THENPRINT"[RVS]DISK ERROR:[OFF]"E;E$,E1;E2
3050 RETURN
READY.
```

DISK FILE LOGGER : JIM BUTTERFIELD

```
100 DIMT$(4)
110 PRINT"DISK FILE LOG - JIM BUTTERFIELD"
120 DATA 169,0,162,4,149,95,202,16,251,169,160,133,94,162,2,32,198,255
130 DATA 230,96,208,10,230,97,208,6,230,96,208,2,230,95,32,228,255,165,150
140 DATA 240,235,32,204,255,198,94,6,98,38,97,38,96,38,95,16,244,96
150 FORJ=860TO914:READX:POKEJ,X:NEXTJ
160 DATA "XXX","SEQ","PRG","USR","REL"
170 FORJ=0TO4:READT$(J):NEXT
180 INPUT"PRINTER";Z$
190 Z=3:IFASC(Z$)=89THENZ=4:INPUT"DATE  #[3CL]";D$
200 INPUT"FILE CHECK";Z$
210 IFASC(Z$)=89THENF=1
220 OPEN4,Z:OPEN1,8,15,"I0":CLOSE1
230 G$="                "
240 OPEN1,8,3,"$0"
250 FORJ=1TO142:GET#1,A$:NEXTJ
260 FORJ=1TO16:GET#1,A$:IFA$=CHR$(160)GOTO280
270 N$=N$+A$
280 NEXTJ:GET#1,A$,A$
290 FORJ=1TO2:GET#1,A$:IFA$=CHR$(160)GOTO310
300 I$=I$+A$
310 NEXTJ
320 FORJ=1TO92:GET#1,A$:NEXTJ
330 PRINT#4,"*** DISK LOG ***  ";D$
340 PRINT#4,"DISK: ";N$;"   ID: ";I$
350 M=M+1:GET#1,K$,T$,S$:IFS$=""THENS$=CHR$(0)
360 F$="":FORJ=1TO16:GET#1,A$:IFA$=CHR$(160)GOTO380
370 F$=F$+A$
380 NEXTJ
390 FORJ=1TO10:GET#1,A$:NEXTJ
400 L=0:IFA$<>""THENL=ASC(A$)
410 GET#1,A$:IFM<8THENGET#1,A$,A$:GOTO430
420 M=0
430 SW=ST:IFK$=""GOTO640
440 K=ASC(K$)-128:IFK<1ORK>4THENK=0
450 IFZ=3ANDF=1GOTO480
460 H$=MID$(STR$(ASC(T$)),2):PRINT#4,LEFT$(G$,3-LEN(H$));H$;
470 H$=MID$(STR$(ASC(S$)),2):PRINT#4,LEFT$(G$,3-LEN(H$));H$;" ";
480 PRINT#4,T$(K);
490 H$=MID$(STR$(L),2):PRINT#4,LEFT$(G$,4-LEN(H$));H$;" ";
500 PRINT#4,F$;LEFT$(G$,17-LEN(F$));
510 IFK=0GOTO630
520 IFK=2ORF=1THENOPEN2,8,4,"0:"+F$+","+T$(K)+",R"
530 A=0:IFK>2GOTO570
540 GET#2,A$,B$:A=0:IFA$<>""THENA=ASC(A$)
550 B=0:IFB$<>""THENB=ASC(B$)
560 GOSUB700
570 IFF=0GOTO620
580 POKE1,92:POKE2,3:A=A+USR(0)
590 IFK<>2THENPRINT#4,A;"BYTES";:GOTO620
600 PRINT#4," ";:AX=A/256:A=A-AX*256:B=B+AX
610 GOSUB700
```

```
620 CLOSE2
630 PRINT#4
640 IFSW=0GOTO350
650 CLOSE1:CLOSE4:END
700 X=B/16:GOSUB710:X=A/16
710 FORJ=1TO2:X%=X:X=(X-X%)*16:IFX%>9THENX%=X%+7
720 PRINT#4,CHR$(X%+48);:NEXTJ:RETURN
READY.
```

```
610 PRINT"[CU]":IFS1<9ANDP$=" "THENP=128-P:GOTO490
620 IFP$="Y"THENT=A(0):S=A(1):GOTO390
630 GOTO370
640 REM
650 INPUT#15,E,E$,E1,E2:IFE=0THENRETURN
660 PRINT"[RVS]DISK ERROR:[OFF]"E;E$,E1;E2:END
670 V=V/16:FORL=1TO2:V%=V:V=(V-V%)*16:IFV%>9THENV%=V%+7
680 PRINT#4,CHR$(V%+48);:NEXTL:RETURN
READY.
```

DISK VIEW : JIM BUTTERFIELD

```
100 PRINT"[CLR,CD]DISK SECTOR VIEWER - JIM BUTTERFIELD"
110 FORJ=30TO90:IFPEEK(32768+J)=32THENNEXTJ:STOP
120 S1=J/5
130 DIM B(77,4),A(255)
140 B$=CHR$(17):INPUT"DRIVE#";D$:IFD$="S"THEND$="0":B$=CHR$(3)
145 IFD$<>"0"ANDD$<>"1"GOTO140
150 OPEN15,8,15,"I"+D$:GOSUB640
160 OPEN3,8,3,"#"+D$:GOSUB640
170 Z$=CHR$(0):GET#3,A$:A=ASC(A$+Z$)
180 IFA=10RA=65THEND1=1:T9=35:S9=3
190 IFA=67THEND1=257:T9=77:S9=4
200 IFT9=0THENCLOSE3:PRINT"?? DISK NOT RECOGNIZED ??":STOP
210 INPUT"PRINTER";P$:IFASC(P$)<>8900TO230
220 OPEN4,4:PRINT#4:S1=16:GOTO240
230 OPEN4,3
240 S2=S1*16-1:FORJ=1TOD1:GET#3,A$:NEXTJ
250 PRINT#4,"      FREE BLOCK MAP"
260 FORJ=1TOT9:T1=0:PRINT#4
270 IFJ=51THENGET#3,A$,A$,A$,A$
280 GET#3,A$:C=ASC(A$+Z$)
290 PRINT#4,RIGHT$(" "+STR$(J),2);" ";
300 FORK=0TOS9-1:GET#3,A$:A=ASC(A$+Z$)
310 B(J,K)=A:FORL=0TO7:D1=32:A%=A/2:IFA<>A%*2THEND1=42:T1=T1+1
320 A=A%:PRINT#4,CHR$(D1);:NEXTL,K
330 IFT1<>CTHENPRINT#4,"?";
340 NEXTJ
350 CLOSE3:PRINT#4:PRINT#4:PRINT#4:PRINT
360 OPEN2,8,2,"#0":GOSUB640
370 PRINT"[HOME,21CD]■TRACK,SECTOR  0,0[5CL]";
380 INPUTT,S
390 IFT<1ORT>T9THENCLOSE4:CLOSE2:CLOSE15:END
400 PRINT"[RVS] WORKING "
410 S%=S/8:S5=S-S%*8:S6=INT(21*S5+.5):T6=B(T,S%)
420 F$="[ALLOCATED]":IFT6ANDS6THENF$="[FREE]"
430 PRINT#15,"B-R:2,";D$;T;S:GOSUB640
440 FORJ=0TO255
450 PRINT#15,"M-R";CHR$(J);B$
460 GET#15,A$:IFA$=""THENA$=CHR$(0)
470 A(J)=ASC(A$):NEXTJ
480 P=0
490 PRINT"[CLR]TRACK";T;"SECTOR";S;F$
500 IFS>8THENPRINT#4:PRINT#4:PRINT#4,"TRACK";T;"SECTOR";S;F$
510 FORJ=PTOP+S2STEPS1:PRINT#4
520 V=J:GOSUB670:PRINT#4,": ";:FORK=0TOS1-1:V=A(J+K):GOSUB670:PRINT#4," ";
530 NEXT K:PRINT#4,":";:FORK=0TOS1-1:V=A(J+K):IF(VAND127)<32THENV=32
540 PRINT#4,CHR$(V);:NEXTK,J
550 PRINT#4:IFS1<9THENPRINT"[RVS] PRESS SPACE FOR OTHER BYTES"
560 PRINT"NEXT TRACK,SECTOR: ";
570 IFA(0)=0THENPRINT"NONE":A$="QUIT":GOTO590
580 PRINTA(0);A(1):A$="GO THERE"
590 PRINTA$;"?";
600 GETP$:IFP$=""GOTO600
```

DISK MOD : JIM BUTTERFIELD

```
100 PRINT"[CLR,CD]DISK VIEWER/CHANGER     JIM BUTTERFIELD"
110 PRINT"[CD]    CAUTION - USE CARE - THIS PROGRAM"
120 PRINT"        CAN WRECK YOUR DISK IF USED"
130 PRINT"      WITHOUT CARE & UNDERSTANDING!"
140 FORJ=20TO85:IFPEEK(32768+J)<>32GOTO160
150 NEXTJ:STOP
160 L1=J:S1=L1/5
170 S2=S1*16-1:S3=5+S1*3
180 DIM A(255)
190 B$=CHR$(17):INPUT"DRIVE#";D$:IFD$="S"THEND$="0":B$=CHR$(3)
200 IFD$<>"0"ANDD$<>"1"GOTO190
210 OPEN 15,8,15,"I"+D$:GOSUB500
220 OPEN2,8,2,"#0":GOSUB500
230 PRINT"[HOME,20CD]■W TRACK,SECTOR  0,0[5CL]";
240 INPUTT,S
250 IFT<1ORT>77THENCLOSE2:CLOSE15:END
260 PRINT"[RVS] WORKING ":PRINT#15,"B-R:2,";D$;T;S:GOSUB500
270 PRINT#15,"B-R:2,";D$;T;S:GOSUB500
280 FORJ=0TO255
290 PRINT#15,"M-R";CHR$(J);B$
300 GET#15,A$:IFA$=""THENA$=CHR$(0)
310 A(J)=ASC(A$):NEXTJ
320 P=0
330 PRINT"[CLR]TRACK";T;"SECTOR";S
340 FORJ=PTOP+S2STEPS1:PRINT:PRINT"]";
350 V=J:GOSUB800:K$="":PRINT"- ";:FORK=0TOS1-1:V=A(J+K):GOSUB800:PRINT" ";
360 NEXT K:PRINT"-";:FORK=0TOS1-1:V=A(J+K):IF(VAND127)<32THENV=32
370 V2=VAND63:IFV2=44ORV2=58ORV2=34THENV=32
380 PRINTCHR$(V);:NEXTK,J
390 PRINT:IFS1<9THENPRINT "[RVS]S[OFF]WAP";
400 PRINT:PRINT"NEXT TRACK,SECTOR: ";
410 IFA(0)=0THENPRINT"NONE":A$="QUIT":GOTO430
420 PRINTA(0);A(1):A$="GO THERE"
430 PRINTA$;"?";
440 INPUTC$:Z=ASC(C$)
450 PRINT"[CU]":IFS1<9ANDZ=83THENP=128-P:GOTO330
460 IFZ=89THENT=A(0):S=A(1):GOTO250
470 IFZ=93GOTO530
480 GOTO230
500 REM
510 INPUT#15,E,E$,E1,E2:IFE=0THENRETURN
520 PRINT"[RVS]DISK ERROR:[OFF]"E;E$,E1;E2:END:RETURN
530 IFLEN(C$)<S3GOTO230
540 IFMID$(C$,4,1)<>"-"ORMID$(C$,S3+1,1)<>"-"GOTO230
550 C1=2:GOSUB700
560 C3=C2-1:FORK=1TOS1:C1=K*3+3:GOSUB700
570 PRINT#15,"M-W";CHR$(C3+K);B$;CHR$(1);CHR$(C2)
580 NEXTK
590 PRINT#15,"U2:2,";D$;T;S:GOSUB500
600 GOTO270
700 C2=0:FORJ=0TO1:C%=ASC(MID$(C$,C1+J)):IFC%<58THENC%=C%-48
710 IFC%>64THENC%=C%-55
```


microfacts

- Sales and Purchase Ledgers
- Open Item accounts
- Nominal and VAT Ledgers
- Invoice preparation
- Stock recording & control
- Job Costing
- Fully Integrated
- Large Capacities
- One pair of Disks only
- Multi Companies
- Powerful & Flexible
- Simple to Operate

MMS
Computer Systems
MMS Ltd., Ketwell House, 75-79 Tavistock Street, Bedford MK40 2RR.
Tel. (0234) 40601 Telex 826311

Basic Programs

```
720 IFC%<@0RC%>15THENSTOP
730 C2=C2*16+C%:NEXTJ:RETURN
800 V=V/16:FORL=1TO2
810 V%=V:V=(V-V%)*16:IFV%>9THENV%=V%+7
820 K$=K$+CHR$(V1)
830 PRINTCHR$(V%+48);:NEXTL:RETURN
READY.



DIRECTORY ALPHABETIZER - BASIC 4


100 REM ****************************
110 REM * DIRECTORY ALPHABETIZER   *
120 REM *                          *
130 REM *       P.GABOR            *
140 REM *              11/03/82    *
150 REM ****************************
160 :
170 C1=0:C2=1
180 DIM E1$(500), TS(50,1)
190 AH$=CHR$(17)
200 TS(1,0)=39: TS(1,1)=1:                REM    FIRST DIRECTORY TRACK/SECTOR
210 AZ$=CHR$(0): BZ$=AZ$+AZ$+AZ$+AZ$
220 FOR K=1TO6 : AZ$=AZ$+AZ$: NEXT
230 AZ#=LEFT$(AZ$,27)
240 GOTO2000
270 REM ****************************
280 REM * BUFFER-READ & ERROR CHK  *
290 REM ****************************
295 PRINT#15,"B-R:"3;DD$;T;S
300 IF DS<20 THEN RETURN
310 PRINTDS$
320 CLOSE3: CLOSE15: END
330 :
370 REM ****************************
380 REM * OPEN FILE & GET BUFFER   *
390 REM ****************************
400 OPEN15,8,15,"I"+DD$: GOSUB300
410 OPEN3,8,3,"#0": GOSUB300
420 RETURN
430 :
470 REM ****************************
480 REM * READ ONE CHAR FROM BUFFER*
490 REM ****************************
500 PRINT#15,"M-R";CHR$(KK);AH$
510 KK=KK+1
520 GET#15,C$: IF C$="" THEN C$=CHR$(0)
530 RETURN
540 :
570 REM ****************************
580 REM * READ NEXT BLOCK INTO E1$ *
590 REM ****************************
600 T=TS(C2,0): S=TS(C2,1):E=0:           REM     TRACK, SECTOR
610 IF T=0 THEN E=1: RETURN
620 C2=C2+1: KK=0
630 PRINT#15,"B-R:"3;DD$;T;S: GOSUB300
640 FOR K=0 TO 1
650   GOSUB500
660   TS(C2,K)=ASC(C$):                  REM     NEXT TRACK, SECTOR
670 NEXT K
680 FOR J=1 TO 8
690   B$=""
700   GOSUB500: A$=C$: A1=ASC(C$)
710   GOSUB500: A$=A$+C$
720   GOSUB500: A$=A$+C$
730   FOR L=1 TO 27
740     GOSUB500: B$=B$+C$
750   NEXT L
760   KK=KK+2: IF A1<129 THEN 800
770   C1=C1+1: PRINTRIGHT$("    "+STR$(C1),5),
780   PRINTLEFT$(B$,16)
790   E1$(C1)=B$+A$:                     REM     PREPARE STRING FOR SORT
800 NEXT J
810 RETURN
820 :
850 REM ****************************
860 REM * PRINT BUFFER TO DIRECTORY *
870 REM * P1 HAS NR OF ENTRIES      *
880 REM * P2 HAS NR OF BLOCKS PRTED *
890 REM ****************************
900 P2=P2+1: T=TS(P2,0): S=TS(P2,1)
910 PRINT#15,"B-R:"3;DD$;T;S: GOSUB300
920 P=0: FOR K=0TO7
930   P1=P1+1: B$=RIGHT$(E1$(P1),3)
940   P=K*32+2: A$=LEFT$(E1$(P1),27)
950   PRINT#15,"B-P:";3;P
960   IFP1>C1 THEN A$=AZ$: B$=BZ$
970   PRINT#3,B$;A$;
980   N$="   "+STR$(P1): IFP1>C1 THEN1000
990   PRINTRIGHT$(N$,5),LEFT$(A$,16)
1000 NEXT K
1010 PRINT#15,"U2:"3;DD$;T;S
1020 GOSUB300
1030 RETURN
1040 :
1070 REM ****************************
1080 REM *    NO NEXT BLOCK         *
1090 REM ****************************
1100 PRINT#15,"M-W";CHR$(0);AH$;CHR$(2);CHR$(0);CHR$(255)
1110 PRINT#15,"U2:";3;DD$;T;S
1120 GOSUB300
1130 RETURN
1140 :
1170 REM*****************************
1180 REM*     SHELL-METZNER SORT    *
1190 REM*****************************
1200 A3=C1: L=C1
1210 L=INT(L/2):IFL=0THENRETURN
1220 E=1:A2=A3-L
1230 A1=E
1240 P=A1+L
1250 IFLEFT$(E1$(A1),16)<=LEFT$(E1$(P),16)THEN1300
1260 A$=E1$(A1): E1$(A1)=E1$(P)
1270 E1$(P)=A$: A1=A1-L
1280 IFA1<1THEN1300
1290 GOTO1240
1300 E=E+1
1310 IFE>A2THEN1210
1320 GOTO1230
1950 :
1970 REM ****************************
1980 REM *   M A I N  P R G M       *
1990 REM ****************************
2000 PRINT"□                                   "
2010 PRINT"       ■■ DIRECTORY ALPHABETIZER ■■"
2040 PRINT
2280 PRINTTAB(25)"P.GABOR"
2290 PRINT"■_______"
2300 PRINT"■WARNING!■     POWER FAILURE DURING"
2310 PRINT"              PROCESS MAY DAMAGE DISK.■"
2320 PRINT"              BACKUP COPY SHOULD BE"
2330 PRINT"              AVAILABLE!■■"
2340 INPUT"DRIVE #";DD$
2350 IF DD$<>"0" AND DD$<>"1" THEN 2340
2500 GOSUB400
2510 PRINT"□          DIRECTORY LISTING:"
2520 PRINT
2530 GOSUB600
2540 IF E=0 THEN2530
2550 :
2560 PRINT: POKE633,0: SYS4541,E1$(0)
2570 PRINT:PRINT"                                  "
2580 PRINT"          ■■ NEW DIRECTORY ■■"
2590 PRINT
2600 P1=0: P2=0
2610 GOSUB900
2620 IF P1<C1 THEN2610
2630 GOSUB1100
2640 FOR K=P2+1 TO C2-1:                  REM      FREE BLOCKS
2650   IF P2>C2-2 THEN 2720
2660   T=TS(K,0): S=TS(K,1)
2670   GOSUB295
2680   GOSUB1100
2690   PRINT#15,"B-F:";DD$;T;S
2700   GOSUB300
2710 NEXT K
2720 CLOSE3: CLOSE15
2730 PRINT"□"
2740 DIRECTORYD(VAL(DD$))
2750 END
READY.



?
.
.  1187 30 00 91 11 96 0A 82 20
.  118F 4B 00 9D 11 A0 0A A0 33
.  1197 3A 20 A0 31 35 00 A6 11
.  119F AA 0A 99 22 93 22 00 B5
.  11A7 11 B4 0A DA 44 28 C5 28
.  11AF 44 44 24 29 29 00 BB 11
.  11B7 BE 0A 80 00 00 00 20 F5
.  11BF BE 20 2B C1 A5 07 F0 4E
.  11C7 38 A5 44 E9 02 85 C2 A5
.  11CF 45 E9 00 85 C3 38 A0 01
.  11D7 B1 C2 E9 01 85 5E 88 B1
.  11DF C2 E9 00 85 5F A5 44 85
.  11E7 C2 A5 45 85 C3 A0 00 84
.  11EF C0 84 C1 18 A5 C2 69 03
.  11F7 85 C2 90 02 E6 C3 B1 C2
.  11FF F0 17 E6 C0 D0 02 E6 C1
.  1207 A5 C1 C5 5F 90 06 A5 C0
.  120F C5 5E B0 05 90 DD 4C 00
.  1217 BF F0 43 A9 00 85 C0 85
.  121F C1 18 A5 44 69 03 85 44
.  1227 90 02 E6 45 A0 00 B1 44
.  122F F0 2B E6 C0 D0 02 E6 C1
.  1237 A6 C1 E4 5F 90 06 A6 C0
.  123F E4 5E B0 19 C8 71 44 85
.  1247 C2 C8 B1 44 69 00 85 C3
.  124F 88 A5 45 91 C2 88 A5 44
.  1257 91 C2 B8 50 C4 60 A5 C0
.  125F 85 BC A5 C1 85 BD 46 BD
.  1267 66 BC D0 04 A5 BD F0 AB
.  126F A9 01 85 B6 A9 00 85 B7
.  1277 38 A5 C0 E5 BC 85 B8 A5
.  127F C1 E5 BD 85 B9 A5 B6 85
.  1287 B1 A5 B7 85 B2 18 A5 B1
.  128F 65 BC 85 BA A5 B2 65 BD
.  1297 85 BB A9 00 F0 02 90 C6
.  129F A5 B1 0A AA A5 B2 2A A8
.  12A7 18 8A 65 B1 AA 98 65 B2
.  12AF A8 18 8A 65 44 85 60 98
.  12B7 65 45 85 61 A5 BA 0A AA
.  12BF A5 BB 2A A8 18 8A 65 BA
.  12C7 AA 98 65 BB A8 18 8A 65
.  12CF 44 85 62 98 65 45 85 63
.  12D7 A0 02 B1 60 85 67 B1 62
.  12DF 85 69 88 B1 60 85 66 B1
.  12E7 62 85 68 88 B1 60 D1 62
.  12EF 90 02 B1 62 AA A0 FF D0
.  12F7 08 90 8A B0 88 90 9F D0
.  12FF 8C C8 B1 66 29 7F 85 C2
.  1307 B1 68 29 7F C5 C2 F0 10
.  130F 08 AD 79 02 F0 05 28 B0
.  1317 49 D0 1C 28 90 44 D0 17
.  131F CA D0 DE A0 00 B1 62 D1
.  1327 60 08 AD 79 02 F0 05 28
.  132F B0 30 90 03 28 90 2B E6
.  1337 B6 D0 02 E6 B7 A5 B9 C5
.  133F B7 D0 04 A5 B8 C5 B6 B0
.  1347 B2 90 B2 38 A5 B1 E5 BC
.  134F 85 B1 A5 B2 E5 BD 85 B2
.  1357 90 DD D0 A3 A5 B1 F0 D7
.  135F D0 9D A0 02 B1 60 AA B1
.  1367 62 91 60 8A 91 62 88 B1
.  136F 60 AA B1 62 91 60 8A 91
.  1377 62 88 B1 60 AA B1 62 91
.  137F 60 8A 91 62 88 D0 C4 00
.  1387 AA AA AA AA AA AA AA AA
.
.
READY.


                LIFE EXPECTANCY


 10 GOTO1000
 100 POKE143,0:WAIT143,64:POKE158,0:PRINT"[5CD,3CR,RVS]ANSWER:[OFF] ";:POKE167,0
:M$=""
 105 WAIT158,1:GETG$
 110 IFMANDG$="Y"THENPRINT"YES":GOTO190
 120 IFMANDG$="N"THENPRINT"NO":M=0:GOTO190
 130 IFMTHEN105
 140 M=ASC(G$):IFM=13THENM=VAL(M$):GOTO190
 150 IFM=20ANDM$>""THENPRINT" [2CL] [CL]";:M$=LEFT$(M$,LEN(M$)-1)
 160 IFM>57ORM<48THEN180
 170 M$=M$+G$:PRINTG$;
 180 WAIT158,1:GETG$:GOTO140
 190 POKE167,1:POKE143,0:PRINT" ":WAIT143,64:RETURN
 1000 POKE59468,14
 1010 PRINT"[CLR]GREETINGS.  I AM GOING TO ASK YOU SOME
 1020 PRINT"[CD]QUESTIONS ABOUT YOURSELF.  THEN, ON THE
 1030 PRINT"[CD]BASIS OF YOUR ANSWERS, I WILL PREDICT
 1040 PRINT"[CD]HOW MANY YEARS YOU WILL LIVE.
 1050 TI$="000000":WAIT142,2
 1060 PRINT"[CLR,CD]HOW OLD ARE YOU?":GOSUB100:A=M
 1070 PRINT"[CLR,CD]ARE YOU FEMALE?":M=1:GOSUB100:S=M
 1080 PRINT"[CLR,CD]ARE YOU A MAORI?":M=1:GOSUB100:R=M
 1090 PRINT"[CLR,CD]HOW MANY MINUTES OF VIGOROUS EXERCISE
 1100 PRINT"[CD]DO YOU HAVE IN A NORMAL WEEK?
 1110 PRINT"[2CD](NOTE:  GOLF, BOWLS AND GARDENING ARE
 1120 PRINT"[CD] NOT[3CL,CD]~~[CU]ACTIVE ENOUGH TO COUNT)":M=0:GOTO100:T=200:M=
M/15-1
 1130 IFM>9THENM=9
 1140 T=T+M*3
 1150 PRINT"[CLR,CD]HOW MANY HOURS OF SLEEP DO YOU NORMALLY
 1160 PRINT"[CD]GET EACH NIGHT?":M=0:GOSUB100:IFM>6ANDMC10THENT=T+4
 1170 PRINT"[CLR,CD]IF YOU COULD MAGICALLY LOSE WEIGHT, HOW"
 1180 PRINT"[CD]MANY POUNDS WOULD YOU LIKE TO LOSE?":M=0:GOSUB100:M=(M-5)/2.5
 1190 IFM>20THENM=20
 1200 T=T-M
 1210 PRINT"[CLR,CD]DO YOU EAT LOTS OF FRESH FRUIT AND"
 1215 PRINT"VEGETABLES?":M=1:GOSUB100:T=T+M*6
 1220 PRINT"[CLR,CD]DO YOU OFTEN HAVE FRIED FOOD  ","[CD](EGGS, CHIPS, MEAT, ETC
.)?
 1230 M=1:GOSUB100:IFM=0THENT=T+8
 1240 PRINT"[CLR,CD]DO YOU SMOKE A PIPE?":M=1:GOSUB100:T=T-M*4
 1250 PRINT"[CLR,CD]ON THE AVERAGE, HOW MANY CIGARETTES"
 1260 PRINT"[CD](OR CIGARS) DO YOU SMOKE PER DAY?":M=0:GOSUB100:IFMTHENM=M+22:C=
M
 1270 T=T-M/3
 1280 PRINT"[CLR,CD]ON THE AVERAGE, HOW MANY DRINKS DO YOU
 1290 PRINT"[CD]HAVE PER DAY?
 1295 PRINT"[3CD]   NOTE:  1 DRINK = 1 BOTTLE OF BEER[9CL,CD,RVS]OR
 1297 PRINTTAB(20)"1 NIP OF SPIRITS[9CL,CD,RVS]OR
```

```
1300 PRINTTAB(20)"1 GLASS OF WINE":M=0:GOSUB100:T=T-4*ABS(M-1)
1310 PRINT"{CLR,CD}HAVE YOU HAD A TETANUS INJECTION IN THE
1320 PRINT"{CD}PAST 10 YEARS?":M=1:GOSUB100:T=T-4+M*4
1330 PRINT"{CLR,CD}IN THE LAST FOUR TIMES YOU WERE IN AN
1340 PRINT"{CD}AUTOMOBILE, HOW MANY OF THOSE TIMES DID
1350 PRINT"{CD}YOU USE THE SEAT BELT?":M=0:GOSUB100:IFM>4THEN1330
1360 T=T+M*2
1370 IFS=0THEN1500
1380 PRINT"{CLR,CD}HAVE YOU HAD A HYSTERECTOMY?":M=1:GOSUB100:IFMTHEN1500
1390 PRINT"{CLR,CD}HAVE YOU HAD A TUBAL LIGATION?":M=1:GOSUB100:IFMTHEN1500
1400 PRINT"{CLR,CD}HAVE YOU REACHED MENOPAUSE?":M=1:GOSUB100:IFMTHEN1500
1410 PRINT"{CLR,CD}ARE YOU PRESENTLY USING BIRTH CONTROL
1420 PRINT"{CD}PILLS?":M=1:GOSUB100:IFMANDC>2THENT=T-10
1425 IFMANDA<36THENT=T+4
1430 PRINT"{CLR,CD}IF BEARING A CHILD WERE TO ENDANGER YOUR
1440 PRINT"HEALTH, WOULD YOU HAVE AN ABORTION?":M=1:GOSUB100:T=T-5+M*6
1450 PRINT"{CLR,CD}DID YOU BEGIN REGULAR SEXUAL ACTIVITY
1460 PRINT"{CD}BEFORE AGE 18?":M=1:GOSUB100:T=T-M
1470 PRINT"{CLR,CD}HAVE YOU BEEN IN CLOSE CONTACT FOR A
1510 PRINT"{CD}YEAR OR MORE WITH SOMEONE HAVING
1520 PRINT"{CD}TUBERCULOSIS?":M=1:GOSUB100:T=T-M*4
1530 PRINT"{CLR,CD}HAVE YOU HAD RADIATION (X-RAY) TREATMENT
1540 PRINT"OF TONSILS, ADENOIDS, ACNE OR RINGWORM?":M=1:GOSUB100:T=T-M*6
1550 PRINT"{CLR,CD}DO YOU REGULARLY WORK WITH ASBESTOS?":M=1:GOSUB100:T=T-M*2
1555 IFMANDCTHENT=T-8
1560 PRINT"{CLR,CD}DO YOU REGULARLY WORK WITH VINYL"
1565 PRINT"CHLORIDE?":M=1:GOSUB100:T=T-M*4
1570 PRINT"{CLR,CD}DO YOU LIVE IN A CITY?":M=1:GOSUB100:T=T-M*6
1580 PRINT"{CLR,CD}HAVE YOU HAD SEXUAL ACTIVITY WITH MORE
1590 PRINT"{CD}THAN TEN DIFFERENT PARTNERS?":M=1:GOSUB100:T=T-M:IFSANDMTHENT=T-
1
1600 PRINT"{CLR,CD}ARE YOU A WIDOW?";:IFS=0THENPRINT"{CL}ER?";
1610 INPUTM:M=1:GOSUB100:T=T-M
1620 PRINT"{CLR,CD}WERE YOU DIVORCED OR SEPARATED IN THE
1630 PRINT"{CD}LAST 12 MONTHS?":M=1:GOSUB100:T=T-M*2
1640 PRINT"{CLR,CD}HAS A CLOSE FAMILY MEMBER (INCLUDING
1650 PRINT"{CD}SPOUSE) DIED IN THE LAST 12 MONTHS?":M=1:GOSUB100:T=T-M
1660 PRINT"{CLR,CD}HAVE YOU HAD A MAJOR ILLNESS OR INJURY
1670 PRINT"{CD}IN THE LAST 12 MONTHS?":M=1:GOSUB100:T=T-M
1680 PRINT"{CLR,CD}HAVE YOU EVER BEEN IN JAIL?":M=1:GOSUB100:T=T-M
1690 PRINT"{CLR,CD}ARE YOU PRESENTLY UNEMPLOYED?":M=1:GOSUB100:T=T-M
1700 PRINT"{CLR,CD}HOW MANY OF YOUR IMMEDIATE FAMILY
1710 PRINT"{CD}(PARENTS, BROTHERS, SISTERS) HAVE DIED
1720 PRINT"{CD}BEFORE AGE 55?":M=0:GOSUB100:T=T-4*M
1730 PRINT"{CLR,CD}HOW MANY OF YOUR MORE DISTANT RELATIVES
1740 PRINT"{CD}(GRANDPARENTS, UNCLES, AUNTS) HAVE DIED
1750 PRINT"{CD}BEFORE AGE 55?":M=0:GOSUB100:T=T-M
1760 IFS=0THEN1800
1770 PRINT"{CLR,CD}HAS YOUR MOTHER OR ANY OF YOUR SISTERS
1780 PRINT"{CD}HAD BREAST CANCER?":M=1:GOSUB100:T=T-M*4
1800 PRINT"{CLR,CD}HOW MANY OF YOUR RELATIVES (PARENTS,
1810 PRINT"{CD}BROTHERS, SISTERS, AUNTS, UNCLES,
1820 PRINT"{CD}GRANDPARENTS) HAVE DIABETES?":M=0:GOSUB100:T=T-2*M
2000 DATA60,7,51.2,41.8,32.4,23.5,15.8,9.8,5.5
2010 DATA66.5,56.7,47.37.5,28.4,19.9,12.5,6.8
2020 DATA53.3,44,35.2,26.6,18.9,13,8.7,5
2030 DATA57.1,47.5,38.1,29.2,21.1,14.6,9.9,5.9
2040 IFRTHENFORI=1TO16:READX:NEXT
2050 IFSTHENFORI=1TO8:READX:NEXT
2060 A%=A/10
2070 FORI=1TOA%:READX:NEXT:READY
2072 IFA>80THENX=5:Y=3
2075 M=(10-A+10*A%)/10
2080 A%=T/200*(Y+M*(X-Y))+.5
2090 PRINT"{CLR,4CD}I PREDICT THAT YOU WILL LIVE TO BE
2100 PRINT"{CD}"A+A%"YEARS OLD.
2110 GOTO2110
READY.
```

```
ECONOMICS SIMULATION - BASIC 4.0

 40 POKE2,0
 50 S=.7:B=10000:FORI=1TO9:READC(I):NEXT:I=RND(-TI)
 55 DATA20000,1000,3000,500,2000,1000,2000,15000,60000
 100 POKE59468,14:IFPEEK(144)=85THENPOKE144,88
 110 PRINT"{2SPACE}THE OBJECT OF THIS GAME IS JUST LIKE
 120 PRINT"REAL LIFE:  TO SATISFY YOUR WANTS AND
 130 PRINT"NEEDS.  IN THE GAME YOU DO THIS BY
 140 PRINT"ACCUMULATING AS MANY SATISFACTION POINTS
 150 PRINT"AS YOU CAN.  YOU GET SATISFACTION POINTS
 160 PRINT"BY BUYING CONSUMER GOODS.  YOU GET THE
 170 PRINT"MONEY TO BUY THE CONSUMER GOODS FROM
 180 PRINT"MANAGING A FARM." :GOSUB995
 190 PRINT"{2SPACE}THE GAME WILL LAST FOR 10 TURNS.  AT THE
 200 PRINT"BEGINNING YOU WILL HAVE $10,000 IN THE
 210 PRINT"BANK.  IF YOU MANAGE YOUR FARM WELL,
 220 PRINT"YOU CAN EXPECT TO EARN ABOUT $10,000 PER
 230 PRINT"TURN.  SPEND THIS MONEY WISELY IF YOU
 240 PRINT"WANT TO SURVIVE." :GOSUB995
 250 PRINT"{2SPACE}NOW YOU WILL NOW HAVE THE OPPORTUNITY TO
 260 PRINT"DO SOME SHOPPING.  PRESS THE NUMBER
 270 PRINT"WHICH CORRESPONDS TO THE ITEM YOU WISH
 280 PRINT"TO BUY.  THE COST WILL AUTOMATICALLY BE
 290 PRINT"DEDUCTED FROM YOUR BANK ACCOUNT.  IF YOU
 295 PRINT"WISH TO BUY TWO OF A PARTICULAR ITEM,
 297 PRINT"THEN PRESS ITS NUMBER TWICE.":GOSUB995
 310 PRINT"{2SPACE}CAPITOL GOODS":PRINT"---------------"
 320 PRINT"   1.  TRACTOR"TAB(32)"$ 20,000":IFTTHENGOSUB985
 330 PRINT"   2.  LOAD OF FERTILISER"TAB(35)"1,000
 340 PRINT"CONSUMER GOODS":PRINT"---------------"
 350 PRINT"   3.  FOOD"TAB(32)"$  3,000
 360 PRINT"   4.  CLOTHING"TAB(37)"500
 370 PRINT"   5.  SHELTER"TAB(35)"2,000
 380 PRINT"   6.  WASHING MACHINE"TAB(35)"1,000":IFW1THENGOSUB985
 390 PRINT"   7.  COLOUR T.V."TAB(35)"2,000":IFW2THENGOSUB985
 400 PRINT"   8.  NEW CAR"TAB(34)"15,000":IFW3THENGOSUB985
 410 PRINT"   9.  NEW HOUSE"TAB(34)"60,000":IFW4THENGOSUB985
 415 GOSUB990
 420 PRINT"WHEN YOU HAVE FINISHED SHOPPING PRESS F";:N1=0:N2=0:N3=0
 450 POKE158,0:WAIT158,1:GETG$:IFG$="F"THEN700
 460 G=VAL(G$):IFG=0THEN450
 465 IFC(G)>BTHEN450
 467 B=B-C(G)
 470 ONGOSUB520,540,560,580,600,620,640,660,680:GOSUB990:GOTO450
 520 T=1:RETURN
 540 IFS<1THENS=S+.5:RETURN
 542 S=S+1/(3+S):RETURN
 560 P%=P%+5:N1=1:RETURN
 580 P%=P%+1:N2=1:RETURN
 600 P%=P%+3:N3=1:RETURN
 620 W1=W1+1:P%=P%+5/W1:RETURN
 640 W2=W2+1:P%=P%+8/W2:RETURN
 660 W3=W3+1:P%=P%+60/W3:RETURN
 680 W4=W4+1:P%=P%+200/W4:RETURN
 700 IFN1ANDN2ANDN3THEN720
 705 PRINT"YOU ARE DEAD !":PRINT"YOU CANNOT SURVIVE WITHOUT ";
 710 IFN1=0THENPRINT"FOOD.":GOSUB995:RUN50
 713 IFN2=0THENPRINT"CLOTHING.":GOSUB995:RUN50
 716 PRINT"SHELTER.":GOSUB995:RUN50
 720 IFY=10THEN900
 725 I=1:IFRND(1)>.4THEN800
 730 PRINT"YOUR CROPS ARE INFESTED WITH INSECTS.":GOSUB990
 740 PRINT"HOW MUCH DO YOU WANT TO SPEND ON
 750 PRINT"INSECTICIDE?  ";:X=0:IFB=0THEN780
 753 A#="":POKE167,0
 755 POKE158,0:WAIT158,1:GETG$:IFG$=""ANDA#>""THENA#=LEFT$(A#,LEN(A#)-1):PRINT"
";
 757 IFASC(G$)=13THENPOKE167,1:PRINT" ":GOTO770
 760 IFASC(G$)>57ORASC(G$)<48THEN755
 765 A#=A#+G$:PRINTG$;:GOTO755
 770 X=VAL(A#):IFX>BTHEN730
 780 B=B-X:I=.7+X/3000:IFI>1THENI=1
 800 R=7500*S*I:IFTTHENR=R*2
 810 Y=Y+1:PRINT"END OF YEAR"Y
 820 PRINT"YOUR HARVEST IS WORTH $"INT(R)"."
 830 IFS<1THENPRINT"THE FERTILITY OF YOUR SOIL IS LOW.
 840 IFI<1THENPRINT"YOUR CROP WAS DAMAGED BY INSECTS.
 850 S=S/2:GOSUB995:B=B+R:GOTO310
 900 PRINT"THE GAME IS FINISHED."
 910 PRINT"YOUR SCORE IS"P%"."
 920 PRINT"THAT RATES YOU AS A";:IFP%<120THENPRINT" POOR";:GOTO940
 924 IFP%<240THENPRINT" FAIR";:GOTO940
 928 IFP%<360THENPRINT" GOOD";:GOTO940
 932 PRINT"N EXCELLENT";
 940 PRINT" PLAYER."
 950 TS=298+PEEK(2):IFP%>TSTHENTS=P%:POKE2,TS-298
 960 PRINT"TOP SCORE TODAY IS"TS"." :GOSUB995:RUN50
 985 PRINT"T"TAB(27)"M":RETURN
 990 B=INT(B):POKE216,22:PRINT:PRINT"POINTS:"P%"."; :N=23-LEN(STR$(B)+STR$(P%))
 992 FORI=1TON:PRINT" ";:NEXT:PRINT"IN BANKS $"B:RETURN
 995 POKE216,23:PRINT:PRINT"       PRESS SHIFT TO CONTINUE.";:WAIT152,1:1
 996 WAIT152,1:RETURN
READY.
```


DIPLOMAT

THE PROFESSIONAL COMMUNICATOR

Serial-Parallel-IEEE-Interrupts Calander/Clock-Auto/Load start-Battery backup

A complete I/O and control system on one card with many unique features. All functions controllable from standard Basic using SAVE, LOAD, VERIFY, PRINT, INPUT AND GET.

PARALLEL
EIGHTY I/O lines
2-16 bit timer/counters
handshake lines
8 handshake modes for Centronix etc
(equivalent of 8 PET
user ports)

SERIAL
TWO asynchronous serial I/O
ports linkable to TTL, RS232C
or current loop 50 to 19,200
baud with 96 byte
extendable buffer

IEEE EXTENSIONS
Device Go-To-Local
Device Clear/Trigger
Serial poll disable/enable
SRQ service request to any
BASIC subroutine

CONTROL FACILITIES
WATCHDOG for hardware and
software errors
1K BYTE BATTERY BACKED
RAM CALANDER/CLOCK
AUTO LOAD/START on power
fail detector

INTERRUPTS
4 edge triggered interrupts jump to any BASIC
subroutine

Diplomats immense interface capability make it particularly suitable for control, data logging, instrumentation, educational and communication applications.

Diplomat is available for all CBM and PET computers and can be mounted internally or externally.

Prices from £380 + VAT (quantity and OEM discounts available).

Send for full details to:—
CORRTRONIC (Technical Services) LTD., BOWDEN HALL, MARPLE, STOCKPORT, CHESHIRE SK6 6ND
Tel. 061-427 7222. Telex. 665905 CORTRO G

Machine Code

"SYSTEM" — Part 2. Handling subroutine parameters.

In part 1, I described how a call-by-name method for subroutines could be implemented on the PET. Having learnt how the basic call mechanism works, we can now go on to see how subroutine parameters are handled.

From listing 1, it can be seen that the sample library, published in part 1, has been extended to include five new subroutines, all of which require the user to specify parameters on the call-line. The first two subroutines, called "BLOCKT" and "XCHNGE", provide the user with some basic functions for manipulating the contents of memory: BLOCKT enables the user to copy, or block transfer, one area of memory to another, and XCHNGE can be used to swap the contents of two memory areas. When calling either of these two routines, the user has to specify, on the call line, the addresses of the memory areas involved, and also the number of bytes to be transferred. The following segment of code shows the call-line syntax needed to invoke the BLOCKT and XCHNGE subroutines:

```
100 REM COPY TOP OF SCREEN TO FIFTH
LINE
110 S=32768          : REM SCREEN START
                       ADDRESS
120 E=32768 +
40*5                 : REM FIFTH LINE AD-
                       DRESS
130 B=40             : REM BYTE COUNT
                       (=LINE LEN)
140 SYSTEM, "BLOCKT", S,E,B

200 REM SWAP TOP TWO SCREEN LINES
210 SYSTEM, "XCHNGE", 32768,
32768+40, 40
```

Note that the example above also demonstrates that parameters can be coded as constants, expressions, or variables. The BLOCKT and XCHNGE routines could be used, for example, to maintain and display several different screen images.

The next two subroutines that are new to the library are called, "READER" and "WRITER". They offer similar functions to those of the 'L' and 'S' commands of the machine language monitor (MLM), and enable the user to store areas of memory as program files on disk. This facility can be particularly useful for frame-handling applications, where the contents of the screen display needs to be recorded on disk for subsequent re-display. The following is an example of how the call-lines for READER and WRITER are coded:

```
100 REM SAVE CURRENT SCREEN
CONTENTS.
110 SYSTEM, "WRITER", "GO: FRAME 1",
32768, 32768+1000

160 REM RE-DISPLAY FRAMES
170 SYSTEM, "READER", "0 : FRAME 5"
```

Again it should be noted that the parameters can be coded as constants, expressions, or variables. For example, the above call to WRITER could have been coded as:

```
105 N$="1" : S = 32768 : K = 1000
110 SYSTEM,"WRITER","0 :
 FRAME"+N$,S,32768+K
```

The fifth and last member of the new library set is called, "DIRECT", and offers the powerful capability of allowing BASIC expressions to be entered, and evaluated, during program execution. The following example should make the usefulness of the DIRECT routine a little more obvious:

```
100 REM SIMPLE DESK CALCULATOR
110 INPUT "YES SIR",A$
120 IF A$="X" THEN PRINT "BYE SIR" :
STOP
130 SYSTEM,"DIRECT";A$
140 GOTO 110
RUN
YES SIR?A=3
YES SIR?A=4
YES SIR?C=SQR( A*A + B*B)
YES SIR?"PRINT A,B,C"
   3    4    5
YES SIR?X
BYE SIR
```

The DIRECT routine can be put to extremely good use in such applications as, computer aided design, spreadsheet type programs, and

graphing routines, all of which often require computations to be dynamically defined at run time.

From the foregoing examples, we see that specifying parameters is fairly straightforward. The user simply codes a comma after the subroutine name, and follows this with the appropriate number of parameters needed by the particular subroutine. Where there is more than one parameter, each one is separated by a comma.

Let us now turn our attention to the actual mechanism by which the subroutines themselves intercept and evaluate the user-provided parameters. As mentioned twice already, the user can code a parameter in one of three different ways. Consequently, a considerable amount of software is required to evaluate any parameter passed to a subroutine. Fortunately, most of the software needed to evaluate parameters already exists, and is accessable, as subroutines in the PET's ROMS. Therefore the task of evaluating parameters can be reduced to two rather simple procedures, one to handle string parameters, and one to handle numeric parameters. For string parameters, the procedure is as follows. First of all we call a PET ROM routine, which I have named ''NXTFLD'', to scan the subroutine call line for the next parameter. Next, another PET ROM routine, named ''EVALU8'', is called to perform the actual parameter evaluation, such as working out string expressions, or picking up the string from a variable. Finally, yet another PET ROM, named ''STRING'', is called to set up the address of the string in the zero-page locations, $IF and $20, the length of the string being returned in the ''A'' register, or accumulator. The procedure for handling numeric parameters is almost the same as that used for strings. For numerical parameters, the 'NXTFLD'' routine is called to scan the call-line, then EVALU8 is called to evaluate a numeric value. But the final call is not to ''STRING'', but to FIXPNT which ensures that the evaluated result is converted to a fixed point number with the result being stored in the A register (LO byte), and the Y register (HI byte). The result can also be found in zero-page locations $11, and $12, containing the LO and HI byte values respectively. Both of the procedures described can be more readily understood by referring to the sample library listing. While writing the subroutines, I have attempted to keep the coding simple, and provided, hopefully, sufficient annotation to enable interested readers to extend the library further.

Listing 2 shows a short BASIC program which can be used to test the five new library routines.

Finally, instead of having to enter the source code by hand, a disk is available which contains the source code as shown, plus some other useful library routines. The disk costs 10 pounds and can be ordered by writing to me, Danny Doyle, at:
9 Main Street, Gawcott, Bucks.

Whether you choose to buy the disk, or decide to enter the code by hand, I very much hope that you will find the ''SYSTEM'' approach to subroutine calling as useful as I have.

System

```
DISPA.SRC.....PAGE 0001

LINE# LOC   CODE        LINE

0001  0000            ;
0002  0000            ; *********************************
0003  0000            ; *                               *  ALL RIGHTS RESERVED
0004  0000            ; *           DECLARE             *
0005  0000            ; *                               *  (C) D. DOYLE L982
0006  0000            ; *    BASIC 4.0 ROM ROUTINES     *  ----------------
0007  0000            ; *                               *
0008  0000            ; *********************************
0009  0000            ;
0010  0000            ERRMSG = $BF00        ; NAME NOT FOUND = 'SYNTAX ERROR'
0011  0000            EVALU8 = $BD98        ; EVALUATES A BASIC EXPRESSION
0012  0000            FIXPNT = $C92D        ; CONVERTS TO FIXED POINT
0013  0000            FLGEND = $BAD2        ; MARKS END OF BASIC INPUT BUFF
0014  0000            NXTFLD = $BEF5        ; SCANS PAST A COMMA
0015  0000            PRGLOD = $F356        ; PERFORMS 'LOAD'
0016  0000            PRGSAV = $F6E3        ; PERFORMS 'SAVE'
0017  0000            SCAN   = $0070      ; STATEMENT SCANNER - CHARGET
0018  0000            STRING = $C7B8        ; SETS UP CHARACTER STRING PNTR
0019  0000            TOKENS = $B4FB        ; BASIC KEYWORD TOKENISER
0020  0000            XECUTE = $B785        ; EXECUTES BASIC STATEMENT
0021  0000            ;
0022  0000            ; *********************************
0023  0000            ; *                               *
0024  0000            ; *         DECLARE               *
0025  0000            ; *                               *
0026  0000            ; *    CONSTANTS, VARIABLES       *
0027  0000            ; *                               *
0028  0000            ; *       AND POINTERS            *
0029  0000            ; *                               *
0030  0000            ; *********************************
0031  0000            ;
0032  0000            BIADLO =  0000          ; BASIC INPUT BUFFER ADDR LO
0033  0000            BIADHI =  0002          ;          DITTO          HI
0034  0000            CASE   = $E84C        ; CASE-MODE SWITCH-ADDRESS
0035  0000            COLON  = $003A        ; ASCII VALUE FOR A COLON
0036  0000            DEVICE = $00D4        ; CELL TO HOLD DEVICE CODE
0037  0000            DISK   =  0008        ; DISK DEVICE CODE
0038  0000            END    = $00C9        ; POINTER TO PROGRAM END
0039  0000            ENTPNT = $0000        ; SUBROUTINE START ADDRESS PNTR
0040  0000            ENTLEN =  0008        ; ENTRY SIZE
0041  0000            FILNOM = $00DA        ; FILENAME POINTER
0042  0000            FNADHI = $0007        ; MLM PTR TO FN FOR SAVE/LOAD
0043  0000            FNADLO = $0002        ; MLM PTR TO FN FOR SAVE/LOAD
0044  0000            FNMLEN = $00D1        ; CELL HOLDS FILENAME LENGTH
0045  0000            IMGPTR = $0077        ; CHARGET IMAGE POINTER
0046  0000            LODVFY = $009D        ; LOAD OR VERIFY FLAG
0047  0000            LOWER  =  0014      ; LOWER-CASE MODE VALUE
0048  0000            NAMSIZ =  0006        ; SUBROUTINE NAME SIZE
0049  0000            START  = $00FB      ; POINTER TO PROGRAM START
0050  0000            STATUS = $0096        ; I/O STATUS FLAG
0051  0000            SUBNAM = $001F        ; NAME ON 'SYSTEM' CALL
0052  0000            TABNAM = $0000        ; POINTER TO CURRENT E.P.T. NAME
0053  0000            UPPER  =  0012      ; UPPER-CASE MODE VALUE
0054  0000            VARPTR = $001F        ; CURRENT VARIABLE NAME
0055  0000            ZERO   = $0000        ; ZERO = 0. SEEMS REASONABLE!


DISPA.SRC.....PAGE 0002

LINE# LOC   CODE        LINE

0056  0000            WREG0  = $005E        ; WORKING REGISTER
0057  0000            WREG1  = $0060        ; WORKING REGISTER
0058  0000            WREG2  = $0011        ; WORKING REGISTER
0059  0000            ;
0060  0000            ;
0061  0000            ;
0062  0000            ; *********************************
0063  0000            ; *                               *
0064  0000            ; *                               *
0065  0000            ; * [] D I S P A T C H E R  []    *
0066  0000            ; *                               *
0067  0000            ; * THIS ROUTINE ENABLES BASIC    *
0068  0000            ; * PROGRAMS TO CALL-BY-NAME      *
0069  0000            ; * MACHINE-CODE SUBROUTINES.     *
0070  0000            ; *                               *
0071  0000            ; *  E.G.                         *
0072  0000            ; *                               *
0073  0000            ; * 10 SYSTEM,"CASELO"            *
0074  0000            ; *                               *
0075  0000            ; * 90 SYSTEM,"CASEUP"            *
0076  0000            ; *                               *
0077  0000            ; *********************************
0078  0000            ;
```

Machine Code

```
THE DISPATCHER......PAGE 0003

LINE# LOC   CODE        LINE

0080  0000                    *=$6000
0081  6000                ;
0082  6000  D8            SYSTEM CLD              ; WE DON'T WANT DECIMAL MODE
0083  6001  20 F5 BE             JSR NXTFLD       ; SCAN TO START OF SUBRTN NAME
0084  6004  20 98 BD             JSR EVALUS       ; EVALUATE CHARACTER STRING
0085  6007  20 B8 C7             JSR STRING       ; THEN SET UP POINTER IN SUBNAM
0086  600A  A9 3F                LDA #EPTLEN      ; GET THE LENGTH OF THE E.P.T
0087  600C  4A                   LSR A            ; DIVIDE E.P.T LENGTH BY EIGHT
0088  600D  4A                   LSR A            ; (NUMBER OF BYTES PER ENTRY)
0089  600E  4A                   LSR A            ; YIELDS NUMBER OF ENTRIES
0090  600F  AA                   TAX              ; ENTRY COUNT IS LOOP CONTROL
0091  6010  A0 00                LDY #ZERO        ; INITIALISE CHARACTER INDEX
0092  6012  A9 42                LDA #<EPTAB      ; SET UP E.P.T NAME POINTER LO
0093  6014  85 00                STA TABNAM       ;
0094  6016  A9 60                LDA #>EPTAB      ; SAME FOR NAME POINTER HI
0095  6018  85 01                STA TABNAM+1     ;
0096  601A  B1 1F       NXTCHR LDA (SUBNAM),Y    ; GET NEXT CHAR OF CALL NAME
0097  601C  D1 00                CMP (TABNAM),Y   ; SAME AS E.P.T NAME?
0098  601E  D0 0D                BNE NXTENT       ; IF NOT, THEN BRANCH TO NXTENT
0099  6020  C8                   INY              ; BUMP CHARACTER POINTER
0100  6021  C0 06                CPY #NAMSIZ      ; HAVE WE MATCHED ALL CHARS
0101  6023  D0 F5                BNE NXTCHR       ; IF NOT, GO MATCH NEXT CHAR
0102  6025                ;
0103  6025                ; ********************************
0104  6025                ; *                              *
0105  6025                ; * CONTROL WILL COME HERE WHEN  *
0106  6025                ; * WE SUCCESSFULLY 'LOOK-UP'    *
0107  6025                ; * THE SUBROUTINE NAME.         *
0108  6025                ; *                              *
0109  6025                ; ********************************
0110  6025                ;
0111  6025  B1 00                LDA (ENTPNT),Y   ; LO BYTE OF SUBRTN ENTRY-POINT
0112  6027  48                   PHA              ; AND STACK IT
0113  6028  C8                   INY              ; BUMP TO THE HI BYTE
0114  6029  B1 00                LDA (ENTPNT),Y   ; NOW THE HI BYTE
0115  602B  48                   PHA              ; GETS STACKED
0116  602C                ;
0117  602C                ; NOW THE STACK HAS BEEN PRIMED. SO THE FOLLOWING
0118  602C                ; RTS INSTRUCTION WILL PASS CONTROL TO THE RQUESTED
0119  602C                ; SUBROUTINE.
0120  602C                ;
0121  602C  60                   RTS              ; D I S P A T C H E D !
```

```
THE DISPATCHER......PAGE 0004
LINE# LOC   CODE        LINE

0123  602D                ;
0124  602D                ; ********************************
0125  602D                ; *                              *
0126  602D                ; * CONTROL COMES HERE WHEN THE  *
0127  602D                ; * CALL NAME AND E.P.T. NAME    *
0128  602D                ; * FAIL TO MATCH. IF THERE ARE  *
0129  602D                ; * MORE NAMES IN THE E.P.T WE   *
0130  602D                ; * TRY AGAIN, ELSE IT'S RATHER  *
0131  602D                ; * BAD NEWS FOR THE USER!       *
0132  602D                ; *                              *
0133  602D                ; ********************************
0134  602D  A0 00       NXTENT LDY #ZERO          ; RE-INITIALISE CHARACTER PNTR
0135  602F  A5 00              LDA TABNAM         ; COMPUTE ADDRESS OF NEXT ENTRY
0136  6031  18                CLC                ; PREPARE FOR ADDITION
0137  6032  69 08              ADC #ENTLEN        ; ADD ENTRY LENGTH
0138  6034  85 00              STA TABNAM         ; SAVE BACK NEW ADDRESS
0139  6036  A5 01              LDA TABNAM+1       ; MAYBE WE HAD A CARRY
0140  6038  69 00              ADC #ZERO          ; SO CATER FOR THE POSSIBILITY
0141  603A  85 01              STA TABNAM+1       ; JUST IN CASE
0142  603C  CA                 DEX                ; ONE LESS ENTRY TO COMPARE
0143  603D  D0 DB              BNE NXTCHR         ; BRANCH IF MORE ENTRIES
0144  603F                ;
0145  603F                ; ********************************
0146  603F                ; *                              *
0147  603F                ; * CONTROL COMES HERE IF THE    *
0148  603F                ; * CALLED SUBROUTINE DOES NOT   *
0149  603F                ; * EXIST, IN WHICH CASE THE     *
0150  603F                ; * INTREPID USER WILL BE GIVEN  *
0151  603F                ; * BASIC'S VERSION OF THE       *
0152  603F                ; * TWO-FINGERED SALUTE, I.E     *
0153  603F                ; * THE MESSAGE 'SYNTAX ERROR'   *
0154  603F                ; * WILL BE DISPLAYED, AND THE   *
0155  603F                ; * LITTLE OL' PROGGY WILL COME  *
0156  603F                ; * TO A GRINDING HALT IN TRUE   *
0157  603F                ; * USER-FRIENDLY FASHION!       *
0158  603F                ; *                              *
0159  603F                ; ********************************
0160  603F                ;
0161  603F  20 00 BF             JSR ERRMSG        ; 'ON YER BIKE!'
```

```
E. P. T.......PAGE 0005

LINE# LOC   CODE        LINE

0163  6042                ;
0164  6042                ; ********************************
0165  6042                ; *                              *
0166  6042                ; * HERE IS THE DISPATCHER'S     *
0167  6042                ; * ENTRY-POINT-TABLE (E.P.T)    *
0168  6042                ; *                              *
0169  6042                ; * ENTRIES ARE EIGHT BYTES LONG *
0170  6042                ; * AND CONSIST OF A 6-CHARACTER *
0171  6042                ; * SPACE-FILLED NAME, FOLLOWED  *
0172  6042                ; * BY THE ENTRY ADDRESS MINUS 1 *
0173  6042                ; * (WHICH GETS PUSHED ONTO THE  *
0174  6042                ; * STACK FOR A CRAFTY RTS JUMP  *
0175  6042                ; *                              *
0176  6042                ; ********************************
0177  6042                ;
0178  6042                ;
0179  6042  43 41       EPTAB  .BYTE   'CASELO'   ;
0180  6048  61 FF              .DBYTE   CASELO-1   ; SWITCH TO LOWER CASE ROUTINE
0181  604A  43 41              .BYTE   'CASEUP'   ;
0182  6050  62 05              .DBYTE   CASEUP-1   ; SWITCH TO UPPER CASE ROUTINE
0183  6052  42 4C              .BYTE   'BLOCKT'   ;
0184  6058  62 0C              .DBYTE   BLOCKT-1   ; MEMORY BLOCK TRANSFER ROUTIN
E
0185  605A  58 43              .BYTE   'XCHNGE'   ;
0186  6060  62 0F              .DBYTE   XCHNGE-1   ; MEMORY BLOCK TRANSFER ROUTIN
E
0187  6062  52 45              .BYTE   'READER'   ;
0188  6068  62 70              .DBYTE   READER-1   ; XFER DISK TO MEMORY ROUTINE
0189  606A  57 52              .BYTE   'WRITER'   ;
0190  6070  62 78              .DBYTE   WRITER-1   ; XFER MEMORY TO DISK ROUTINE
0191  6072  44 49              .BYTE   'DIRECT'   ;
0192  6078  62 B4              .DBYTE   DIRECT-1   ; DIRECT MODE INPUT - D I M P
0193  607A                ;
0194  607A                ; FUTURE ENTRIES CAN BE DEFINED HERE
0195  607A                ;
0196  607A                EPTLEN = *-EPTAB          ; LENGTH OF E.P.T.
```

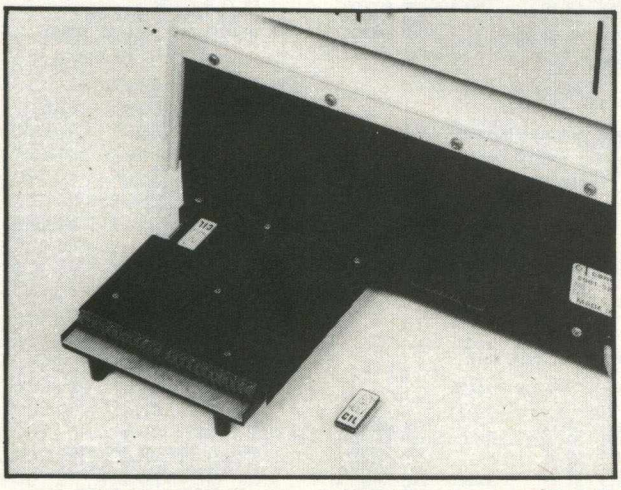
```
THE LIBRARY......PAGE 0006

LINE# LOC   CODE        LINE

0198  607A                ;
0199  607A                ; WARNING - IF 'EPTLEN' ADDRESS IS > THAN $61FF
3200  607A                ; THEN ADJUST THE FOLLOWING LIBRARY START ADDRESS
0201  607A                ;
0202  607A                ; ********************
0203  607A                ; * START LIBRARY ON *
0204  607A                ; *                  *
0205  607A                      *=$6200
0206  6200                ; *                  *
0207  6200                ; *   PAGE BOUNDARY  *
0208  6200                ; ********************
0209  6200                ;
0210  6200                ; ********************************
0211  6200                ; *                              *
0212  6200                ; *       L I B R A R Y          *
0213  6200                ; *                              *
0214  6200                ; *     R O U T I N E S          *
0215  6200                ; *                              *
0216  6200                ; *         S T A R T            *
0217  6200                ; *                              *
0218  6200                ; *          H E R E             *
0219  6200                ; *                              *
0220  6200                ; ********************************
0221  6200                ;
0222  6200                ;       ***************
0223  6200                ;       *             *
0224  6200                ;       * C A S E L O *
0225  6200                ;       *             *
0226  6200                ;       ***************
0227  6200                ;
0228  6200  A9 0E       CASELO LDA #LOWER
0229  6202  8D 4C E8             STA CASE
0230  6205  60                   RTS
0231  6206                ;
0232  6206                ;       ***************
0233  6206                ;       *             *
0234  6206                ;       * C A S E U P *
0235  6206                ;       *             *
0236  6206                ;       ***************
0237  6206                ;
0238  6206  A9 0C       CASEUP LDA #UPPER
0239  6208  8D 4C E8             STA CASE
0240  620B  60                   RTS
0241  620C                ;
0242  620C                ;       ***************
0243  620C                ;       *             *
0244  620C                ;       * X C H N G E *
0245  620C                ;       *   &         *
0246  620C                ;       * B L O C K T *
0247  620C                ;       *             *
0248  620C                ;       ***************
0249  620C                ;
0250  620C                ; X C H N G E: XCHNGE IS USED TO SWAP TWO MEMORY AREAS.
0251  620C                ; -----------
0252  620C                ;         CALLED VIA: SYSTEM,"XCHNGE",S1,S2,N
```

```
THE LIBRARY......PAGE 0007

LINE# LOC   CODE        LINE

0253  620C                ;
0254  620C                ; WHERE S1 AND S2 ARE THE START ADDRESSES OF THE TWO
0255  620C                ; AREAS TO BE SWAPPED, AND N IS THE BYTE COUNT.
0256  620C                ;
0257  620C                ; B L O C K T: BLOCKT IS MEMORY BLOCK MOVE ROUTINE
0258  620C                ; -----------
0259  620C                ;
0260  620C                ;         CALLED VIA: SYSTEM,"BLOCKT",S1,S2,N
0261  620C                ;
0262  620C                ; WHERE S1 AND S2 CONTAIN THE "FROM" AND "TO"
0263  620C                ; ADDRESSES, AND N IS THE BYTE COUNT.
0264  620C                ;
0265  620C  01         XBFLAG .BYTE      1       ; XCHNGE/BLOCKT CALL FLAG
0266  620D  CE 0C 62     BLOCKT DEC XBFLAG       ; FLAG BLOCKT CALL
0267  6210  78          XCHNGE SEI              ; ADD 'GO FASTER' STRIPES!
0268  6211  20 67 62             JSR XCHARG      ; GET AREA 1 ADDRESS
0269  6214  48                   ;HA             ; AREA 1 ADDRESS TO STACK
0270  6215  98                   TYA             ;
0271  6216  48                   PHA             ;
0272  6217  20 67 62             JSR XCHARG      ; GET AREA 2 ADDRESS
0273  621A  48                   PHA             ; AREA 2 ADDRESS TO STACK
0274  621B  98                   TYA             ;
0275  621C  48                   PHA             ;
0276  621D  20 67 62             JSR XCHARG      ; BYTE COUNT ENDS UP IN WREG2
0277  6220  68                   PLA             ; RETRIEVE THE AREA 1 ADDRESS
0278  6221  85 60                STA WREG1       ;
0279  6223  68                   PLA             ;
0280  6224  85 61                STA WREG1+1     ; SAVE IN A WORKING REGISTER
0281  6226  68                   PLA             ;
0282  6227  85 5E                STA WREG0       ; SAME FO THE AREA 2 ADDRESS
0283  6229  68                   PLA             ;
0284  622A  85 5F                STA WREG0+1     ;
0285  622C  A0 00                LDY #ZERO       ; INITIALISE AREA INDEX
0286  622E  A5 12       XCHG0    LDA WREG2+1     ; BYTE COUNT = 0 IF FINISHED
0287  6230  D0 04                BNE XCHG0       ; BRANCH IF STILL MORE TO SWAP
0288  6232  A5 11                LDA WREG2       ; HAVE WE FINISHED FOR CERTAIN?
0289  6234  F0 2A                BEQ XCHGX       ; IF YES, BRANCH TO EXIT
0290  6236  B1 5E       XCHG0    LDA (WREG0),Y   ; GET THE NEXT AREA 1 BYTE
0291  6238  AE 0C 62             LDX XBFLAG      ; TEST XCHNGE OR BLOCKT CALL
0292  623B  F0 06                BEQ AREAMV      ; IF BLOCKT THEN SKIP THE SWAP
0293  623D  48                   PHA             ; IF XCHNGE, STACK AREA 1 BYTE
0294  623E  B1 60                LDA (WREG1),Y   ; THEN GET THE AREA 2 BYTE
0295  6240  91 5E                STA (WREG0),Y   ; BECOMES THE NEW AREA 1 BYTE
0296  6242  68                   PLA             ; POP THE OLD AREA 1 BYTE
0297  6243  91 60       AREAMV   STA (WREG1),Y   ; BECOMES THE NEW AREA 2 BYTE
0298  6245                ;
0299  6245                ; NOW WE HAVE EITHER SWAPPED, OR BLOCK-TRANSFERRED,
0300  6245                ; A BYTE. NEXT WE UPDATE THE MEMORY AREA POINTERS
0301  6245                ; AND BYTE COUNT, AND PREPARE FOR THE NEXT BYTE.
0302  6245                ;
0303  6245  38                   SEC             ; PREPARE FOR SUBTRACTION
0304  6246  A5 11                LDA WREG2       ; GET THE BYTE COUNT
0305  6248  E9 01                SBC #1          ; ONE LESS BYTE TO DO
0306  624A  85 11                STA WREG2       ; STORE RECOMPUTED BYTE COUNT
0307  624C  A5 12                LDA WREG2+1     ; COMPLETE THE COMPUTATION
```

A NOTICE TO ALL

PET LOVERS

INTRODUCE YOUR PETS TO OURS
AND SOLVE YOUR INTERFACE PROBLEMS



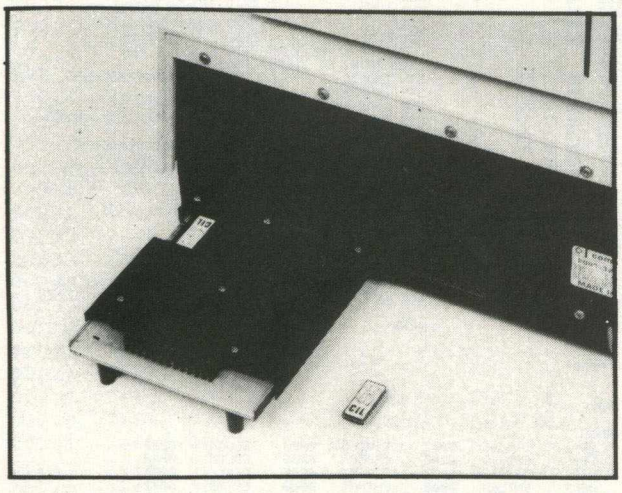
Only £195.00

ANALOGUE/DIGITAL I/O

THE PUPI

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

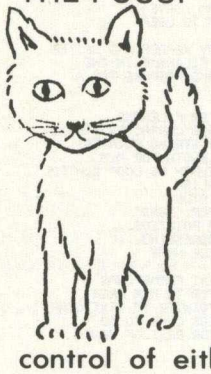
When connected to the "PET" User Port the PUPI gives you all the above features together with an operating system in EPROM, which interacts with Basic's variables, giving extremely simple operation. Inputs and outputs are ±10V and relays are rated at 10VA. Logic inputs can be used for microswitch sensing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



Only £195.00

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

Machine Code

```
THE LIBRARY......PAGE 0008

LINE# LOC   CODE        LINE


0308  624E  E9 00              SBC #ZERO           ;
0309  6250  85 12              STA WREG2+1         ;
0310  6252  E6 5E              INC WREG0           ; BUMP AREA 1 ADDRESS BY 1
0311  6254  D0 02              BNE XCHG1           ;
0312  6256  E6 5F              INC WREG0+1         ; CATER FOR CARRY
0313  6258  E6 60    XCHG1     INC WREG1           ; BUMP AREA 2 ADDRESS BY 1
0314  625A  D0 D2              BNE XCHG            ; GO AND HANDLE ANOTHER BYTE
0315  625C  E6 61              INC WREG1+1         ; OR HANDLE CARRY
0316  625E  D0 CE              BNE XCHG            ; AND THEN GO FOR THE NEXT BYTE
0317  6260              ;
0318  6260              ; AT THIS POINT ALL BYTES HAVE BEEN PROCESSED. ALL THAT
0319  6260              ; REMAINS TO BE DONE IS TO ALLOW INTERRUPTS, AND RESET
0320  6260              ; THE XCHNGE/BLOCKT FLAG.
0321  6260              ;
0322  6260  58         XCHGX  CLI                 ; OFF WITH GO-FASTER STRIPES
0323  6261  A9 01              LDA #1              ; ASSUME XCHNGE CALL COMING
0324  6263  8D 0C 62           STA XBFLAG          ; AND PRIME THE FLAG BYTE
0325  6266  60                 RTS                 ; RETURN TO USER
0326  6267              ;
0327  6267              ; THE FOLLOWING ROUTINE IS USED BY XCHNGE, BLOCKT,
0328  6267              ; AND WRITER TO EVALUATE 'SYSTEM' CALL PARAMETERS.
0329  6267              ;
0330  6267  20 F5 BE    XCHARG JSR NXTFLD          ; SCAN FOR NEXT ARGUMENT
0331  626A  20 98 BD           JSR EVALU8          ; EVALUATE THE EXPRESSION
0332  626D  20 2D C9           JSR FIXPNT          ; CONVERT TO INTEGER VALUE
0333  6270  60                 RTS
0334  6271              ;
0335  6271              ;       ***************
0336  6271              ;       *             *
0337  6271              ;       * R E A D E R *
0338  6271              ;       *     &       *
0339  6271              ;       * W R I T E R *
0340  6271              ;       *             *
0341  6271              ;       ***************
0342  6271              ;
0343  6271              ; R E A D E R: READER PROVIDES DISK TO MEMORY TRANSFERS.
0344  6271              ; -----------
0345  6271              ;
0346  6271              ;           CALLED VIA: SYSTEM,"READER","FILENAME"
0347  6271              ;
0348  6271              ; WHERE FILENAME IS IN STANDARD DOS FORMAT
0349  6271              ;
0350  6271              ; W R I T E R: WRITER PROVIDES MEMORY TO DISK TRANSFERS
0351  6271              ; -----------
0352  6271              ;           CALLED VIA: SYSTEM,"WRITER","FILENAME",S,E
0353  6271              ;
0354  6271              ; WHERE FILENAME IS IN STANDARD DOS FORMAT, S IS THE
0355  6271              ; START ADDRESS, AND E IS THE END ADDRESS OF THE
0356  6271              ; AREA TO BE SAVED
0357  6271              ;
0358  6271  20 92 62    READER JSR SETFIL          ; SET UP FILENAME FOR I/O
0359  6274  20 56 F3           JSR PRGLOD          ; PERFORM 'LOAD'
0360  6277  EA                 NOP                 ; POINT TO CHECH I/O STATUS
0361  6278  60                 RTS                 ; RETURN TO USER
0362  6279              ;

THE LIBRARY......PAGE 0009

LINE# LOC   CODE        LINE


0363  6279  20 92 62    WRITER JSR SETFIL          ; SET UP FILENAME FOR I/O
0364  627C  A9 08              LDA #DISK           ; GET THE DEVICE CODE
0365  627E  85 D4              STA DEVICE          ; AND TELL OS WHAT IT IS
0366  6280  20 67 62           JSR XCHARG          ; GET THE START ADDRESS
0367  6283  84 FB              STY START           ; AND TELL OS WHERE IT IS
0368  6285  85 FC              STA START+1         ;
0369  6287  20 67 62           JSR XCHARG          ; GET THE END ADDRESS
0370  628A  84 C9              STY END             ; AND TELL OS WHERE IT IS
0371  628C  85 CA              STA END+1           ;
0372  628E  20 E3 F6           JSR PRGSAV          ; PERFORM 'SAVE'
0373  6291  60                 RTS                 ; RETURN TO USER
0374  6292              ;
0375  6292              ; THE FOLLOWING ROUTINE IS USED BY READER AND WRITER
0376  6292              ; TO ESTABLISH THE USER-PROVIDED FILENAME IN THE
0377  6292              ; MACHINE-LANGUAGE-MONITOR AREA AND PREPARE FOR A
0378  6292              ; SAVE OR LOAD COMMAND.
0379  6292              ;
0380  6292  20 F5 BE    SETFIL JSR NXTFLD          ; SCAN FOR FILENAME
0381  6295  20 98 BD           JSR EVALU8          ; EVALUATE CHARACTER STRING
0382  6298  20 B8 C7           JSR STRING          ; SET UP STRING POINTER
0383  629B  85 D1              STA FNMLEN          ; SAVE LENGTH FOR MLM
0384  629D  AA                 TAX                 ; USE LENGTH AS LOOP CONTROL
0385  629E  A0 00              LDY #ZERO           ; INITIALISE
0386  62A0  84 96              STY STATUS          ; I/O STATUS
0387  62A2  84 9D              STY LODVFY          ; FLAG FOR 'LOAD'
0388  62A4  A9 07              LDA #FNADHI         ; SET UP POINTERS...
0389  62A6  85 DA              STA FILNOM          ; FOR ADDRESSING...
0390  62A8  A9 02              LDA #FNADLO         ; THE FILE NAME
0391  62AA  85 DB              STA FILNOM+1        ;
0392  62AC  B1 1F    XFRFNM LDA (VARPTR),Y        ; GET NEXT FNAME CHAR
0393  62AE  91 DA              STA (FILNOM),Y      ; TRANSFER TO MLM AREA
0394  62B0  C8                 INY                 ; BUMP POINTER TO NEXT CHAR
0395  62B1  CA                 DEX                 ; ONE LESS CHAR TO DO
0396  62B2  D0 F8              BNE XFRFNM          ; LOOP FOR REST OF NAME
0397  62B4  60                 RTS                 ; RETURN
0398  62B5              ;
0399  62B5              ;       ***************
0400  62B5              ;       *             *
0401  62B5              ;       * D I R E C T *
0402  62B5              ;       *             *
0403  62B5              ;       ***************
0404  62B5              ;
0405  62B5              ; D I R E C T: DIRECT ALLOWS EXECUTION OF BASIC DIRECT
0406  62B5              ; ----------- MODE STATEMENTS.
0407  62B5              ;
0408  62B5  ----------  ;           CALLED VIA: SYSTEM,"DIRECT",E
0409  62B5              ;
0410  62B5              ; WHERE E IS A STRING, ENCLOSED IN QOUTES, OR A STRING
0411  62B5              ; VARIABLE CONTAINING THE DIRECT-MODE STATEMENT.
0412  62B5              ;
0413  62B5  20 F5 BE    DIRECT JSR NXTFLD          ; SCAN FOR STATEMENT
0414  62B8  20 98 BD           JSR EVALU8          ; EVALUATE CHARACTER STRING
0415  62BB  20 B8 C7           JSR STRING          ; SET UP STRING POINTERS
0416  62BE  AA                 TAX                 ; SAVE LENGTH FOR LOOP
0417  62BF  A9 00              LDA #BIADLO         ; BASIC INPUT BUFFER ADDR

THE LIBRARY......PAGE 0010

LINE# LOC   CODE        LINE


0418  62C1  85 5E              STA WREG0           ;
0419  62C3  A9 02              LDA #BIADHI         ;
0420  62C5  85 5F              STA WREG0+1         ; SAVE IN WORKING REGISTER
0421  62C7  A0 00              LDY #ZERO           ; INITIALISE BUFFER POINTER
0422  62C9  B1 1F    DIR1      LDA (VARPTR),Y      ; GET NEXT STRING CHAR
0423  62CB  91 5E              STA (WREG0),Y       ; INTO BASIC INPUT BUFFER
0424  62CD  C8                 INY                 ; BUMP TO NEXT CHARACTER
0425  62CE  CA                 DEX                 ; ONE LESS TO DO
0426  62CF  D0 F8              BNE DIR1            ; LOOP FOR ALL CHARACTERS
0427  62D1  A5 77              LDA IMGPTR          ; SAVE BASIC SCAN POINTER
0428  62D3  48                 PHA                 ;
0429  62D4  A5 78              LDA IMGPTR+1        ;
0430  62D6  48                 PHA                 ; ONTO THE STACK
0431  62D7  98                 TYA                 ; BASIC INPUT BUFFER PTR
0432  62D8  AA                 TAX                 ; INTO X REGISTER
0433  62D9  20 D2 BA           JSR FLGEND          ; INSERT END MARKER
0434  62DC              ;           SET UP X AND Y REGISTERS
0435  62DC              ;           TO POINT AT DIRECT IMAGE.
0436  62DC  86 77              STX IMGPTR          ; SET UP CHARGET POINTERS
0437  62DE  84 78              STY IMGPTR+1        ;
0438  62E0  20 70 00           JSR SCAN            ; SCAN THE IMAGE
0439  62E3  20 FB B4           JSR TOKENS          ; CONVERT KEYWORDS TO TOKENS
0440  62E6  20 70 00    SCANIT JSR SCAN           ; RE-SCAN
0441  62E9  20 85 B7           JSR XECUTE          ; XQT THE BASIC STATEMENT
0442  62EC  A0 00              LDY #ZERO           ; INITIALISE IMAGE PTR
0443  62EE  B1 77              LDA (IMGPTR),Y      ; GET NEXT CHAR FROM IMAGE
0444  62F0  D0 07              BNE MULTIS          ; BRANCH IF IT MIGHT BE COLON
0445  62F2  68                 PLA                 ; ELSE RESTORE
0446  62F3  85 78              STA IMGPTR+1        ; BASIC IMAGE POINTER
0447  62F5  68                 PLA                 ;
0448  62F6  85 77              STA IMGPTR          ;
0449  62F8  60                 RTS                 ; RETURN TO THE USER
0450  62F9  C9 3A    MULTIS CMP #COLON           ; IS IT A COLON
0451  62FB  F0 E9              BEQ SCANIT          ; IF SO PROCESS NEXT PART
0452  62FD  4C 00 BF           JMP ERRMSG          ; IF NOT, GIVE HIM THE BOOMER
0453  6300              ;
0454  6300                     .END

ERRORS = 0000

SYMBOL TABLE

SYMBOL VALUE
 AREAMV   6243    BIADHI   0002    BIADLO   0000    BLOCKT   620D
 CASE     E84C    CASELO   6200    CASEUP   6206    COLON    003A
 DEVICE   00D4    DIR1     62C9    DIRECT   62B5    DISK     0008
 END      00C9    ENTLEN   0008    ENTPNT   0000    EPTAB    6042
 EPTLEN   003F    ERRMSG   BF00    EVALU8   BD98    FILNOM   00DA
 FIXPNT   C92D    FLGEND   BAD2    FNADHI   0007    FNADLO   0002
 FNMLEN   00B1    IMGPTR   0077    LODVFY   009D    LOWER    000E
 MULTIS   62F9    NAMSIZ   0006    NXTCHR   601A    NXTENT   602D
 NXTFLD   BEF5    PRGLOD   F356    PRGSAV   F6E3    READER   6271

THE LIBRARY......PAGE 0010

LINE# LOC   CODE        LINE


0418  62C1  85 5E              STA WREG0           ;
0419  62C3  A9 02              LDA #BIADHI         ;
0420  62C5  85 5F              STA WREG0+1         ; SAVE IN WORKING REGISTER
0421  62C7  A0 00              LDY #ZERO           ; INITIALISE BUFFER POINTER
0422  62C9  B1 1F    DIR1      LDA (VARPTR),Y      ; GET NEXT STRING CHAR
0423  62CB  91 5E              STA (WREG0),Y       ; INTO BASIC INPUT BUFFER
0424  62CD  C8                 INY                 ; BUMP TO NEXT CHARACTER
0425  62CE  CA                 DEX                 ; ONE LESS TO DO
0426  62CF  D0 F8              BNE DIR1            ; LOOP FOR ALL CHARACTERS
0427  62D1  A5 77              LDA IMGPTR          ; SAVE BASIC SCAN POINTER
0428  62D3  48                 PHA                 ;
0429  62D4  A5 78              LDA IMGPTR+1        ;
0430  62D6  48                 PHA                 ; ONTO THE STACK
0431  62D7  98                 TYA                 ; BASIC INPUT BUFFER PTR
0432  62D8  AA                 TAX                 ; INTO X REGISTER
0433  62D9  20 D2 BA           JSR FLGEND          ; INSERT END MARKER
0434  62DC              ;           SET UP X AND Y REGISTERS
0435  62DC              ;           TO POINT AT DIRECT IMAGE.
0436  62DC  86 77              STX IMGPTR          ; SET UP CHARGET POINTERS
0437  62DE  84 78              STY IMGPTR+1        ;
0438  62E0  20 70 00           JSR SCAN            ; SCAN THE IMAGE
0439  62E3  20 FB B4           JSR TOKENS          ; CONVERT KEYWORDS TO TOKENS
0440  62E6  20 70 00    SCANIT JSR SCAN           ; RE-SCAN
0441  62E9  20 85 B7           JSR XECUTE          ; XQT THE BASIC STATEMENT
0442  62EC  A0 00              LDY #ZERO           ; INITIALISE IMAGE PTR
0443  62EE  B1 77              LDA (IMGPTR),Y      ; GET NEXT CHAR FROM IMAGE
0444  62F0  D0 07              BNE MULTIS          ; BRANCH IF IT MIGHT BE COLON
0445  62F2  68                 PLA                 ; ELSE RESTORE
0446  62F3  85 78              STA IMGPTR+1        ; BASIC IMAGE POINTER
0447  62F5  68                 PLA                 ;
0448  62F6  85 77              STA IMGPTR          ;
0449  62F8  60                 RTS                 ; RETURN TO THE USER
0450  62F9  C9 3A    MULTIS CMP #COLON           ; IS IT A COLON
0451  62FB  F0 E9              BEQ SCANIT          ; IF SO PROCESS NEXT PART
0452  62FD  4C 00 BF           JMP ERRMSG          ; IF NOT, GIVE HIM THE BOOMER
0453  6300              ;
0454  6300                     .END

ERRORS = 0000

SYMBOL TABLE

SYMBOL VALUE
 AREAMV   6243    BIADHI   0002    BIADLO   0000    BLOCKT   620D
 CASE     E84C    CASELO   6200    CASEUP   6206    COLON    003A
 DEVICE   00D4    DIR1     62C9    DIRECT   62B5    DISK     0008
 END      00C9    ENTLEN   0008    ENTPNT   0000    EPTAB    6042
 EPTLEN   003F    ERRMSG   BF00    EVALU8   BD98    FILNOM   00DA
 FIXPNT   C92D    FLGEND   BAD2    FNADHI   0007    FNADLO   0002
 FNMLEN   00B1    IMGPTR   0077    LODVFY   009D    LOWER    000E
 MULTIS   62F9    NAMSIZ   0006    NXTCHR   601A    NXTENT   602D
 NXTFLD   BEF5    PRGLOD   F356    PRGSAV   F6E3    READER   6271



SYMBOL TABLE

SYMBOL VALUE
 SCAN     0070    SCANIT   62E6    SETFIL   6292    START    00FB
 STATUS   0096    STRING   C7B8    SUBNAM   001F    SYSTEM   6000
 TABNAM   0000    TOKENS   B4FB    UPPER    000C    VARPTR   001F
 WREG0    005E    WREG1    0060    WREG2    0011    WRITER   6279
 XBFLAG   620C    XCHARG   6267    XCHG     622E    XCHG0    6236
 XCHG1    6258    XCHGX    6260    XCHNGE   6210    XECUTE   B785
 XFRFNM   62AC    ZERO     0000

END OF ASSEMBLY.
```

```
00180  7eb2                 ;****                  utility routines                  ****
00181  7eb2                 ;************************************************************
00182  7eb2                 ;
00183  7eb2                 ;****************************************
00184  7eb2                 ;****   jump to syntax error routine   ****
00185  7eb2                 ;****************************************
00186  7eb2                 ;
00187  7eb2  20 24 7e       err     jsr getcod                    ;restore chrget code
00188  7eb5  4c 00 bf               jmp synerr                    ;print syntax error
00189  7eb8                 ;
00190  7eb8                 ;****************************************
00191  7eb8                 ;****    check basic line for comma    ****
00192  7eb8                 ;****************************************
00193  7eb8                 ;
00194  7eb8  a9 2c          chkcom lda #44                        ; 44 = ascii ","
00195  7eba  a0 00                 ldy #0                        ;
00196  7ebc  d1 77                 cmp (txtptr),y                ;compare to current
                                                                      basic char
00197  7ebe                 ;
00198  7ebe  d0 f2                 bne err                        ;if not = ","
00199  7ec0                 ;                                        print syntax erro
00200  7ec0  4c 70 00               jmp chrget                    ;get next char
00201  7ec3                 ;
00202  7ec3                 ;****************************************
00203  7ec3                 ;**** conv 2 digit ascii number to hex ****
00204  7ec3                 ;**** store result in temp            ****
00205  7ec3                 ;****************************************
00206  7ec3                 ;
00207  7ec3  38             aschex sec                            ;set carry for sub
00208  7ec4  e9 30                 sbc #48                       ;1st digit to hex
00209  7ec6  8d f5 7e              sta temp                      ;store it in temp
00210  7ec9  20 70 00              jsr chrget                    ;get second digit
00211  7ecc  90 07                 bcc asc1                      ;goto to asc1 if
00212  7ece                 ;                                        btwn 0-9
00213  7ece  c9 2c                 cmp #44                       ;  is it = ","
00214  7ed0  f0 20                 beq ascend                    ;    yes, done
00215  7ed2  4c b2 7e              jmp err                       ;    no, syntax error
00216  7ed5                 ;
00217  7ed5  38             asc1    sec                           ;set carry for sub
00218  7ed6  e9 30                 sbc #48                       ;2nd digit to hex
00219  7ed8  48                    pha                           ;save 2nd digit
00220  7ed9  a9 00                 lda #0                        ;
00221  7edb  ae f5 7e              ldx temp                      ;find tens by mult
00222  7ede  18             asc2    clc                           ;
00223  7edf  69 0a                 adc #10                       ; 1st x 10
00224  7ee1  ca                    dex                           ;
00225  7ee2  d0 fa                 bne asc2                      ;
00226  7ee4  8d f5 7e              sta temp                      ;store tens in temp
00227  7ee7  68                    pla                           ;get 2nd digit
00228  7ee8  18                    clc                           ;
00229  7ee9  6d f5 7e              adc temp                      ;  and add to temp
00230  7eec  8d f5 7e              sta temp                      ;  save it in temp
00231  7eef  20 70 00              jsr chrget                    ;  get next char
00232  7ef2                 ;
00233  7ef2  60             ascend rts                           ;done
00234  7ef3                 ;
00235  7ef3                 ;****************************************
00236  7ef3                 ;****      storage areas             ****
00237  7ef3                 ;****************************************
00238  7ef3                 ;
00239  7ef3  00             ycoord .byte 0                        ;line coordinate
00240  7ef4  00             xcoord .byte 0                        ;column coordinate
00241  7ef5  00             temp   .byte 0                        ;work area
00242  7ef6  00             store  .byte 0,0                      ;storage for code
00242  7ef7  00
00243  7ef8                 ;
```

PAMPER YOUR PET WITH

Codewriter©

"no programming experience necessary"

Write your data base applications in minutes instead of days. Dynatech Microsoftware introduce to PET owners the CODEWRITER, a superb program generator for the 8000 series Pet with 8050 disk drive unit.

Screen layout, data entry validation, screen display of user-defined error messages, screen calculations, searching by any field - all are child's play to Codewriter Disk 1. Codewriter Disk 2 provides printed reports and menu generators.

A selection of the applications of Codewriter already in use are listed below;

Data entry storage and retrieval of;
(a) personnel records
(b) product information
(c) rental records
(d) customer data
Stock control systems
Letter/word information
Hotel reservations systems
Fixed assets and depreciation schedules
Projects records and control systems
Mailing and labelling systems

Turnkey training courses are always available at various Centres throughout the U.K., or take a holiday course in the charming island of Guernsey. For full details on courses and information on Codewriter, write or 'phone NOW!

Dealer enquiries welcome — *we are appointing them fast!*

Codewriter Disk 1 - £125
Codewriter Disk 2 - £65

DYNATECH MICROSOFTWARE LTD.
SUMMERFIELD HOUSE, VALE,
GUERNSEY, CHANNEL ISLAND.
Tel. 0481 47377 Telex 4191130

FREE PET/CBM COMAL

"The excitement in Europe (over COMAL) seems to be growing by the hour and we look forward to America being able to share in the good fortune of having an easy-to-use, structured, planning language at last."

The power of PASCAL and the ease of BASIC can now be yours with Commodore COMAL, a new programming language from DENMARK. It is being distributed in the USA by the COMAL USERS GROUP. To find out more about COMAL and how you can get a free disk copy of Commodore COMAL, send a large self-addressed stamped (35 cents) envelope to:

COMAL USERS GROUP
5501 GROVELAND TER., MADISON, WI 53716.
Outside USA please add $2.00 for airmal and handling.
*PET & CBM are trademarks of Commodore Business Machines.

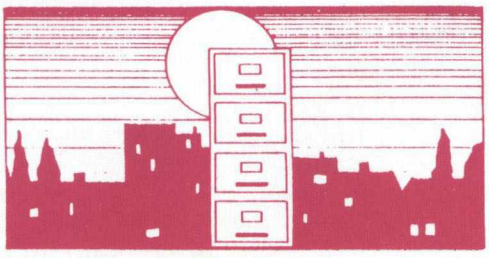
ADVERTISER'S INDEX

RABBIT SOFTWARE

380 STATION ROAD,
HARROW, MIDDX.
HA1 2DE.
Tel: 01-863-0833

RABBIT BASE

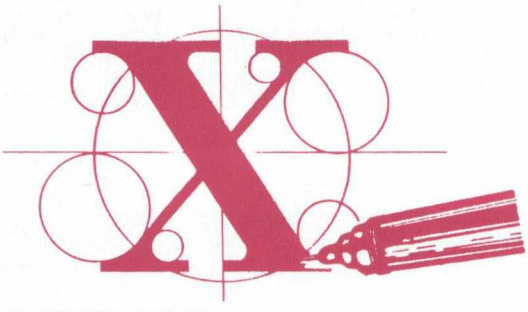
Your own personal Database Management System for your VIC-20. Ideal for any storage and retrieval of any type of record. Powerful search, sort and calculation features.



RETAIL PRICE £14.99

RABBIT WRITER

The wordprocessor for your VIC-20. This program requires 16K RAM Expansion. 'Rabbit Writer' enables the rapid and easy creation of text and its printing to the printer. 'Rabbit Writer' has powerful editing and right hand justification and has many other Word-processing features via the appropriate function keys such as indenting, centering, etc. and enables expanded characters to be printed. It also has the facility to merge i.e. names and addresses from Rabbit Base.

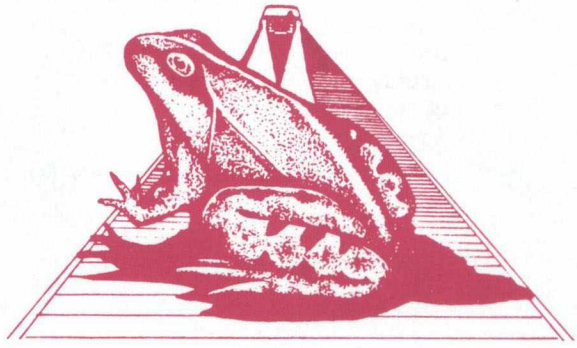


RETAIL PRICE £19.99

NOW ITS HERE !!

FROGGER

Bring the famous pub game home. First cross the motorway without getting run over and then home across logs and turtles. But beware of snakes crocodiles and submerging turtles. Incredible multi-colour mode graphics written in machine code.



RETAIL PRICE £9.99

SPACE STORM

Amazing 3D Graphics of the popular asteroid game. Superb value for money. One of Rabbits best-selling games.

RETAIL PRICE £6.99

Super Worm	£4.99	Rabbit Functions	£4.99	Cosmic Battle	£4.99
Night Flight	£4.99	Ski–Run	£4.99	Rabbit Chase	£4.99
Charpet 20	£4.99	Dune Buggy	£4.99	Code Breaker	£4.99

Available from Vic Dealers or Direct from:

Rabbit Software
380 Station Road,
Harrow, Middlesex HA1 2DE.

Telephone: 01-863 0833
Credit Cards Accepted.

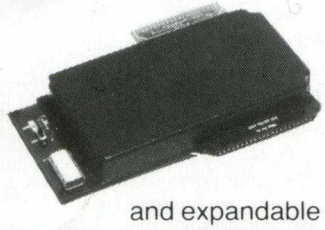
RABBIT SOFTWARE

Please send me _____
I enclose cheque/PO for _____
OR Please debit my Access 5224 _____
Barclaycard 4929 _____ *Expiry date* _____
Name _____
Address _____
_____ *Code* _____
Dealers discount available
ACCESS, BARCLAYCARD ORDERS WELCOME
Mail to: RABBIT SOFTWARE
380 STATION ROAD, HARROW,
MIDDX. HA1 2DE.

VISA

VIC-20 ACCESSORIES FROM STACK

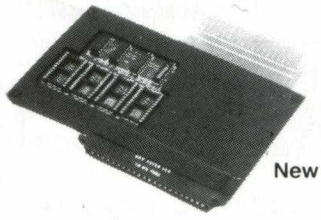
STACK STOREBOARD (memory expansion unit)



Power up your VIC-20 to a MASSIVE 32k COMPUTER!!

only **£49.00** (plus VAT) for 3k and expandable to 27k on the same board.

STACK ROM SWITCHBOARD



Use up to 4 ROMs at once! eg. games. ROMs. VICKIT. VICKIT II etc.

New Lower Price!! **£29.00** (plus VAT)

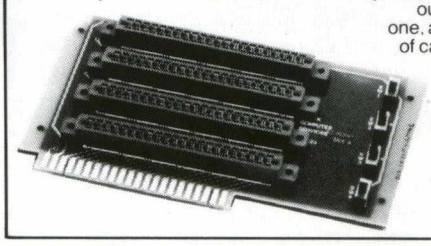
STACK 8k RAMPACK

Use this upgrade pack to increase memory size on Stack Storeboard by 8k a time.

NEW LOWER PRICE!! only £29.00 (plus VAT)

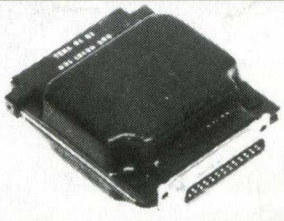
NEW STACK 4-SLOT MOTHERBOARD

Fits into the port at the rear of the Stack Storeboard or the Stack Low Cost 3k Memory and enables you to use up to 4 cartridges in addition to the extra memory on the Storeboard or 3k Memory. Each slot can be switched in or out, thus allowing the use of one, all four, or any combination of cartridges without having to remove them.



only **£24.99** (plus VAT)

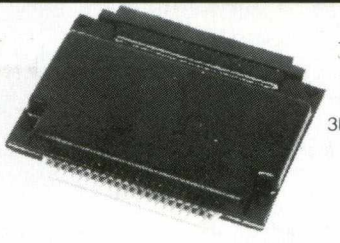
STACK LOW COST RS232 INTERFACE



Allows you to use a serial printer with your VIC-20.

£22.99 (plus VAT)

STACK LOW COST 3k MEMORY



The lowest costing memory addition gives you 6½k of user memory on your VIC-20. Also allows you to use those quality games which demand 3k of Hi-Res Graphics! Socket at rear allows you to stack up further accessories.

only **£25.99** (plus VAT)

STACK LIGHTPEN

Allows you to use VIC-20 without keyboard entry by simple programming. Sensor in pen sees the TV screen! Ideal for education. games. menu selection etc.

only **£25.00** (plus VAT)

STACK VICKIT SERIES

A series of ROMs which greatly simplifies programming and enhance the qualities of your VIC-20. Fits into Stack ROM SWITCHBOARD or Stack STOREBOARD.

VICKIT

Offers HELP to programmers....it also offers AUTO. DELETE. DUMP. FIND. OFF. RENUMBER. STEP. TRACE.

£25.00 (plus VAT)

Special Offer Price if Purchased with STOREBOARD only **£15.00** (plus VAT)

VICKIT II

A 4k ROM offering ALL THE FEATURES OF VICKIT plus.....TEXT. GRAPHICS. LINE. CLEAR. DRAW. PUT. FILL. SET. POINT

only **£29.00** (plus VAT)

Other exciting additions to the VICKIT series due soon!

Contact your local Commodore VIC dealer for details.

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