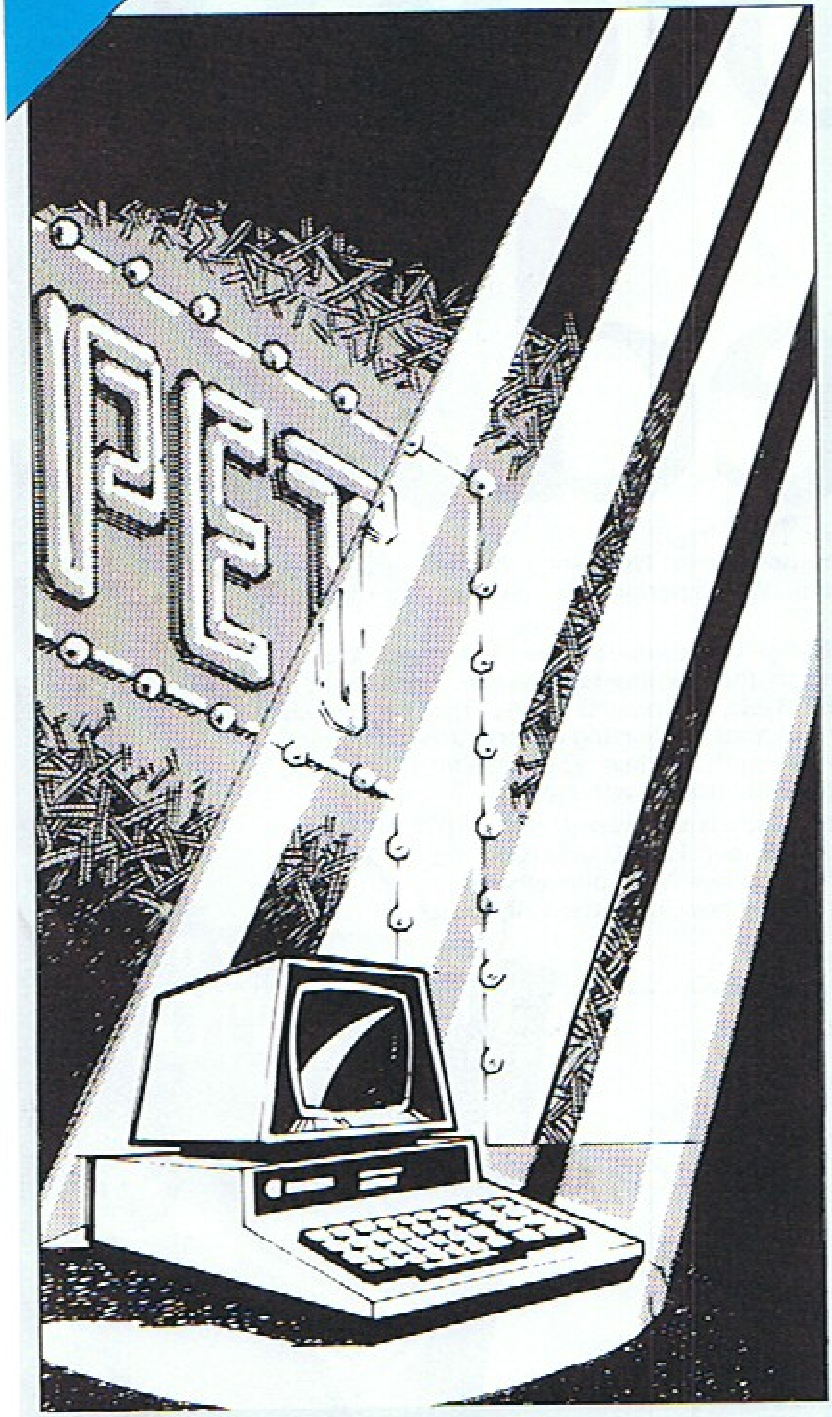
The Second International **PET SHOW** Butterfield Is Here!!!!!!

COMMODORE CLUB NEWS

June 1981

Volume 3 Issue 5



SILICON OFFICE

From the people who brought you OZZ

BUSINESS USER'S COLUMN

More on security

IEEE — 488

A journey on the bus

DISK DRIVES

Part two of our special course

THE PET SHOW

SPECIAL PULL—OUT GUIDE AND CATALOGUE, PLUS **FREE** *ENTRY*

commodore COMPUTER

Contents

THE PET SHOW

Commodore Club News is Published by Commodore Business Machines (U.K.) Ltd., 818, Leigh Road, Slough, Berks and printed by Uniprint, 91-95, Euston Road, London NW1. Whilst the publishers take every care to ensure the accuracy of information contained in "Club News", no liability is accepted for errors or any consequences of such errors that might arise.

Free Entry To PET SHOW

The Second International PET SHOW Butterfield Is Here!!!!!!

June 1981

COMMODORE CLUB NEWS

Volume 3 Issue 5

SILICON OFFICE
From the people who brought you OZZ

BUSINESS USER'S COLUMN
More on security

IEEE — 488
A journey on the bus

DISK DRIVES
Part two of our special course

THE PET SHOW
SPECIAL PULL-OUT GUIDE AND CATALOGUE, PLUS FREE ENTRY

commodore COMPUTER

Please Note:- Only by filling in and presenting this form will *User Club Members* be entitled to free entry for two people to the Second International Pet Show.

Name...

Address...

....

....

Pet User YES / NO

My System is used for...

COMMODORE USER'S CLUB MEMBER — OFFICIAL REGISTRATION FORM

Editorial

Introduction

Welcome to Volume 3 issue 5 of the magazine. You'll have noticed if you saw the last newsletter a significant change in appearance, hopefully for the better. I think this has been a step in the right direction, and that we've continued the trend with this issue. If you disagree you are free, as ever, to contact me and air your views. The Pet Show will be a fine opportunity to do this, and I look forward to seeing many of you there. I'll be on a stand in the front foyer, so you've no excuses! (Neither have I......).

Continuity

With the introduction of a monthly newsletter, one of the things we'll be doing is to introduce an amount of continuity amongst the articles. So, each month you will know that certain features will be appearing. For instance, David Pocock's series on the use of Disk Drives is a regular contribution, and even if you've thought so far that you've known it all, I can guarantee you that David has a number of tricks up his sleeve that you certainly won't have known. We'll be regularly reviewing a number of products, and each month will also see a review of a Commodore Approved Product. This is because we feel that the Approved Product range is a very important part of the Commodore world, and ought to be given as wide a coverage as possible. After all, the Approved Products catalogue contains some of the finest hardware and software available for the Pet.

The next issue will be again featuring the special educational section, as it will be doing every other month, but that doesn't mean that you'll be seeing issues totally devoid of information on the Pet in education. This month there are a number of articles of this nature, and we hope that not only schools and colleges will find these interesting, but the rest of you will as well. There are a vast number of Pets in the educational field - they represent something like 20% to 25% of the Pet population in the U.K. - so we can't afford to ignore you! The list of educational workshops in the last issue has already proved to be of use, and next time we'll be publishing the ones that we missed first time round. This list will be continually updated as the newsletters go by.

This time the special middle section is devoted to the Pet Show, and is really 'everything you've always wanted to know about the Show but were afraid to ask'. Do you approve of this pull-out section policy ? Let me know.

Second-hand Pet Equipment

I've had a number of requests from club members about the availability of second hand Pet equipment. If you know of any source of this, if you've got any Pet equipment that you want to sell (e.g. you've just upgraded to a disc based system from cassettes, and now have a cassette unit to sell, or you've upgraded from Basic 2 to Basic 4 and find yourself with a spare set of Basic 2 roms), or there is an item of hardware that you're after, write to me, with all the details (i.e. item, price and address), at the address at the end of the editorial.

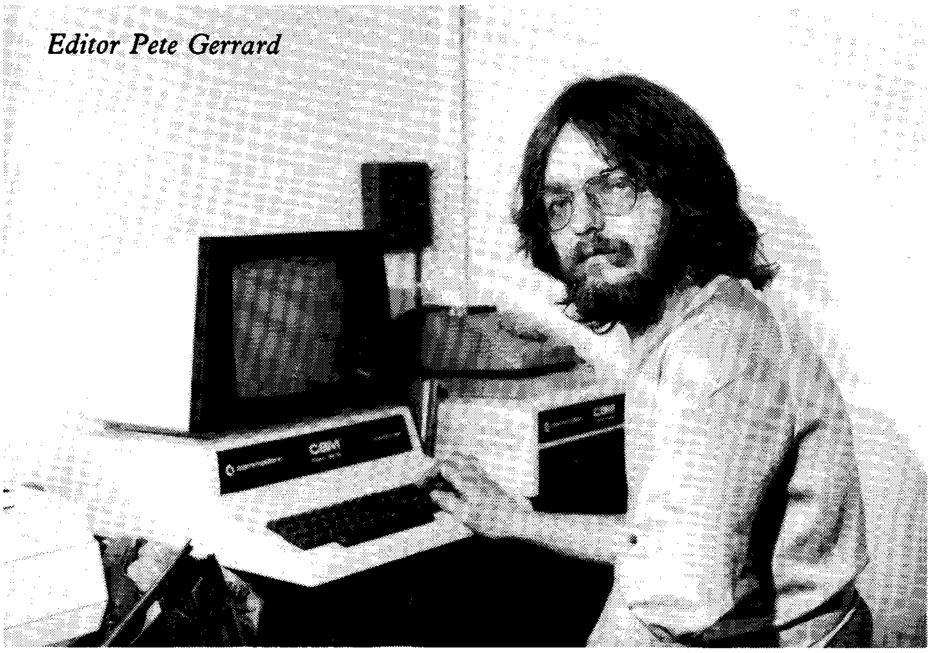
We have available, at a price of £1.00 each, around 700 empty high quality plastic diskette cases. You can buy any number from 1 to 700, but as you can appreciate this is strictly on a first-come-first-served basis. If you're interested in obtaining any of these, write with your cheque made out to C.B.M. (U.K.) Ltd. to Margaret Gulliford at the address below.

Continuing Debate

Finally the continuing debate on what goes into the newsletter. There are a number of requests from people who own 80 column Pets who want to do more than just run their applications software on them. The newsletter has helped in the past, by publishing the various Basic 1, 2 and 4 memory maps, but not everybody is competent enough to convert, alter, play with, whatever programs they've got. I've tried to make amends for this lack of information a little by including in this newsletter an article describing some useful peeks, pokes and chr$ values, but this is really only a begining. So, a plea on behalf of myself, and everybody else who's got an 8032 and wants to know more about it. If you're an 8032 'whizz-kid', and would like to share your learnings with other Pet users, send any contributions to the Editor, at the address below-:

Commodore Business Machines
818 Leigh Road
Trading Estate
Slough
Berks.

If there's ever anything you want to discuss, I'm available on Slough 74111


Editor Pete Gerrard

New Products

Audiogenic

Audiogenic, who supply and make all of Commodore's cassette based software, have just announced a new catalogue of products for the Pet. These are a range of analog and digital interfaces from the company "Connecticut MicroComputer Inc.", and Audiogenic are now the U.K. distributors.

The catalogue should really be called a manual, as it contains much valuable information as well as the product descriptions. In the catalogue you will find such things as a 16 channel analog to digital converter, which converts each of the 16 inputs in turn to an 8 bit digital signal. There is an extremely interesting product known as SADI, which is an interface that allows communication between three devices - an IEEE 488 controller (such as the Pet), a serial RS 232 input/output device, and a parallel output device. This means that, amongst other things, you can have Pets talking to each other. As well as this the catalogue features serial and parallel printer adaptors, and overall is a very useful addition to the Pet world. To obtain a copy of the catalogue, telephone Reading (STD code 0734) 595647. It costs £2.50, offset against your first order, and is really well worth it because it does much more than simply telling you what is available. Described by genial Audiogenic director Martin Maynard as "Cheap at twice the price", I'm inclined to agree with him.

As a footnote, Audiogenic also supply a vast range of books and hardware add-ons for the Pet, and are a good, quick source for such material. A copy of their latest complete catalogue is well worth a browse through, and can be obtained from the same number as above.

Audiogenic Director, Martin Maynard (seated bottom right), with cohorts.

Pets in Space

People have asked me in the past if I knew of any astrological programs for the Pet, or any that calculated planetary orbits for instance. Well, up until now I've been at a loss for an answer, but not any longer! An enterprising company in Michigan, called Matrix, have boldly gone where no company has boldly gone before, and really put the Pet into orbit by producing a lot of astrological programs, amongst other things.

Some of the programs they've got cover such things as the positions (to seconds of arc accuracy) of the Sun, Moon and outer planets, transits and secondary progressions, 10 house systems with planets, and so on. They also publish a comprehensive magazine on all aspects of micros and programming in this particular field. For further information, including a catalogue, write to:-

Matrix Magazine
1014 North Main Street,
Ann Arbor,
Michigan 48104
Ohio
U.S.A.

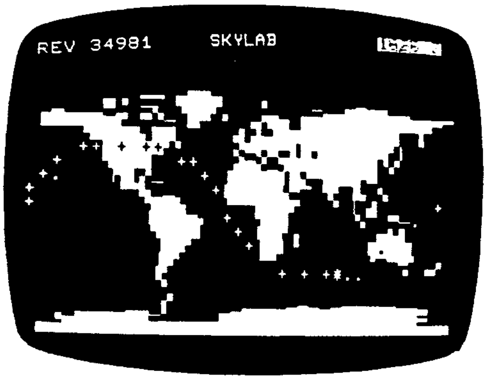
Now, I believe that somewhere in England this company has a U.K. outlet. Can anyone tell me who it is ?

Real Time Orbit

One program that has been generated in the U.K. is one called Real Time Orbit, by G.E. Perry. This program, in Basic, and written for 16k+ 40 column Pets, shows the ground-track and current location of a satellite in real-time.

The internal clock of the Pet is set to Greenwich Mean Time before inputting satellite identity, rev. number, nodal period (in minutes), inclination (in degrees), mean motion (in revs/day from NASA two-line orbital elements), and epoch and longitude of an ascending node for any orbit on the day concerned (in the format of the NASA prediction bulletin).

After computing the current location the program generates a map and plots the ground-track of the current orbit with a cross if the satellite has already passed that point and a dot for future positions (reverse field is used over land masses). Locations coinciding with coastlines are not plotted thereby preserving outline. The current location is indicated by a flashing asterisk. As the satellite moves forward its previous location is replaced by a cross or the appropriate coastline graphic character.



For most of the time a world map is displayed with a ten degree resolution in latitude and longitude. The example above shows the re-entry of Skylab over the south-western corner of Australia on rev 35981 on the 11th of July 1979.

When the satellite reaches Europe, the world map is replaced by a two degree resolution map as displayed above. On leaving this area the world map is regenerated. At the completion of one orbit the world map is regenerated together with the new ground-track. The program works continuously through midnight. There is another version of the program in which the past ground-track is not indicated at all.

Further details are available from:-

G.E. Perry
101 Northampton Road,
Kettering,
Northants NN15 7JY

Educational Plea

A request and a plea for help. The School of Management Studies and Languages, in Buckinghamshire, is looking for tutorial programs in foreign languages (French, Spanish etc.). Hopefully not just straightforward phrase for phrase translation, but if that's what you do have, all well and good. If you use such programs, or know of anywhere that does, could you get in touch with the School at the address below:-

Dr. Tony Bastick,
School of Management Studies and Languages,
Bucks College of Higher Education,
Newland Park,
Gorland Lane,
Chalfont St. Giles,
Bucks.
Or telephone Tony on Chalfont St. Giles 4441. Thanks.

Intex Datalog Compiler

Intex Datalog have just announced the release of their PC-Basic Compiler for Commodore's range of micros. This is written entirely in machine code, and is fully compatible with Commodore Micorsoft interpreted Basic. It supports all Basic 2.0. commands, and all Commodore peripherals. A Basic 4.0 version will be available by the end of September. The compiler takes a program which has been normally saved on disk, and compiles it back on to disk as a directly loadable program file. It can be used with any of the various disk drives, and is used in conjunction with a runtime package in Eprom. The speed of compilation obviously depends on the complexity of the program being compiled, but it is of the order of 50 lines a minute. Increase in speed can be very good - some programs running more than ten times faster. Sale price is £300 plus V.A.T.

For further information, contact Intex Datalog on Eaglescliffe (STD 0642) 781193.

Mannesman Tally

Mannesman Tally, the company who produced Commodore's 8024 printer, are now offering a substantial number of 80 column matrix printers, all of which are now classified as ex-demonstration printers, and these are available at substantially reduced prices. These units have been used and maintained by the company itself, and are offered with a return to factory warranty of 30 days. The prices range from as little as £300 up to £500 for almost new printers. Cash with your order qualifies you for an additional 3% discount, and if you collect youself delivery charges are also deducted. All enquiries about this should be addressed to :-

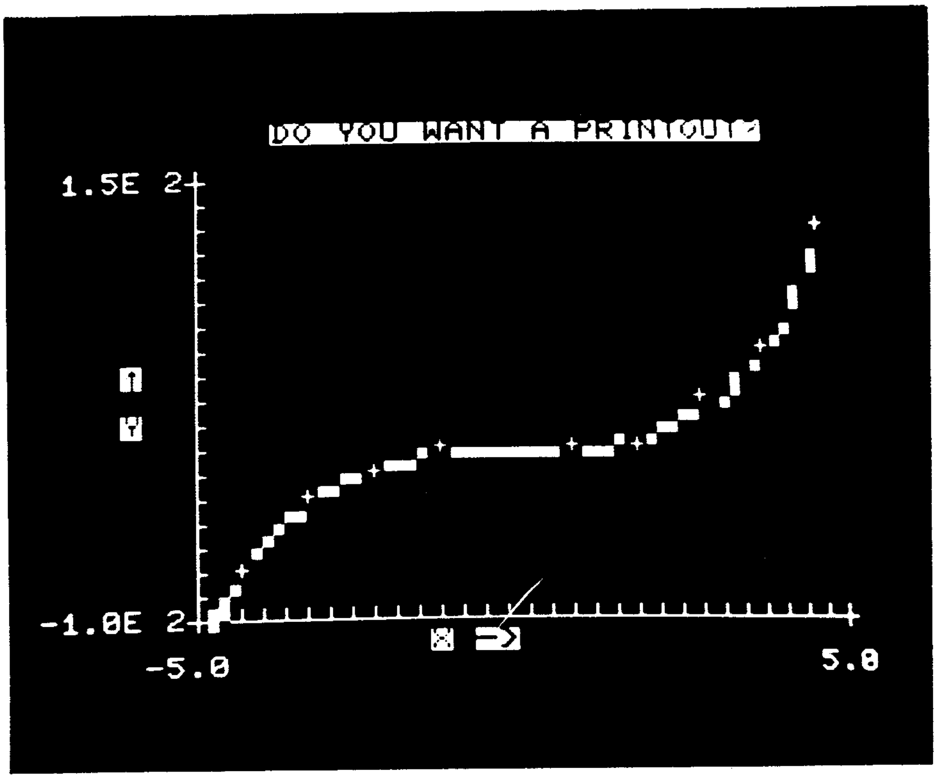
Bernard Lavelle,
Mannesman Tally Ltd.,
(Ex-demo stock)
7 Cremyll Road,
Reading RG1 8NQ

Alternatively you can 'phone Bernard on Reading (STD 0734) 580141 extention 33, quoting "ex-demo stock". This is what I did, and found Bernard a helpful and obliging person. In these inflationary times it might be worth a try.

Tecpacs

I recently attended a press conference, where BHRA Fluid Engineering (developed from the British Hydromechanical Research Association) announced the release of a large number of programs for the Pet. These go under the generic name of Tecpacs, and each program covers a specific topic in a technical area such as Civil Engineering Hydraulics. There are at present 16 programs in the series, all of them being concerned with engineering applications such as Mechanical Engineering, Structural Analysis, and Statistical Data Analysis. All of the

Tecpac System Set-up



DO YOU WANT A PRINTOUT?

Typical screen print

programs are now part of the Commodore Approved range. Time and space forbid too much coverage this time around, but there'll be more in the next issue. Ar the present, for further details contact:-

Technical Software Centre,
BHRA Fluid Engineering,
Cranfield,
Bedford MK43 OAJ
Tel. 0234 750102

A Microcomputer based Record System in an Accident Unit

The following article really proves that Pets get everywhere. It comes from the Accident Unit of the Caernarfonshire and Anglesey General Hospital, in Bangor, Gwynedd, and formed the abstract of a proposed talk by the presenting author Mr. R.H. Gray. We haven't got the full details of the talk, but this does show you what an enterprising lot they are in Bangor.

A microcomputer system for casualty patient records has been developed in the Gwynedd Accident Unit. The hardware consists of standard, readily available business equipment at modest cost. The computer system which has evolved is organised as follows. On admission, direct input of patient information and known details of circumstances concerning the accident and injury is performed by clerical staff. This information is stored on floppy disk and a casualty card printed.

Encoding of medical information is performed by the attending Casulty Officer and at this stage more detailed data on specific "special interest" projects (e.g. Road Traffic Accidents) is put onto the casualty card. This new information is fed into the computer system by the clerical staff and stored. An upgraded hardcopy file is printed on discharge/disposal of the patient. Each patient receives a unique casualty number and retrieval of an individual's file is based on either number or name. The accumulated file is used for routine tasks such as the printing of a daily log book and enquiry into referral of a particular patient.

Also, clinics (Fracture, Hand etc.) can have a print out in individual card form of referred casulty patients. This information can be manipulated to include only the pertinent details relevant to each specialist clinic. This allows the Accident Unit to maintain a master manual hardcopy file of Casualty patient records, simultaneously allowing the main Hospital Records to have a programmable subset of information for each particular clinic.

The use of Basic programming language and involvement of medical personnel directly in program development, combined with hardware that is dedicated solely to use in the Accident Unit, allows marked flexibility. Rapid progress has been achieved with minimal resources. The use of a decentralised system such as this, modified for individual Accident Units, would allow ready compilation of National Statistics.

So, the initiative has been taken -who will follow ?

Silicon Office

Silicon Office is the latest arrival from the Bristol Software Factory, already well known for their earlier programs such as Trader, Item, Monitor, and of course the enormously succesful OZZ, a package which sold over 2,500 copies in the first six months of release. OZZ was the first microcomputer program to begin to exploit the true potential of a small single user system by enabling the user to keep track of information in his own individual way. Silicon Office takes the next step, by covering many office procedures but still keeping the individuality of OZZ. The program will only work on the newly announced 8096 Commodore microcomputer, and will be released concurrently with it at the Pet Show on June 18th.

Silicon Office is constructed from five main building blocks, namely :-

1) An advanced file system for maintaining files of information.

2) A word processor for typing letters, reports etc.

3) A point-to-point communication facility for information transfer.

4) A fourteen digit calculator for mathematical manipulation.

5) Programmibility, to control the system as a whole.

Now you can see why it requires 96k of memory! Not having access to an 8096 I can't give you a detailed review of the product yet - suffice it to say that what I've seen so far is staggering, and in summary Silicon Office is quite simply a revolution in programming. Details are available from the Bristol Software Factory, and their address is :-

Bristol Software Factory
Strahearn House,
88 Queen's Road,
Clifton,
Bristol BS8 1SA
Tel.0272 314278

Pet Paradox

(By Nikki Saunders, and reproduced from the Australian Commodore magazine)

I dedicate this article to the woman whose man has a PET. Not the furry type, or even the lascivious sort, but a cold, calculating machine. My sympathy is extended to every lady whose mate flits away the hours programming his new toy. It really makes you see green!

Life is bad enough when you are confronted with the latest issue of Penthouse, only to be compared with the luscious Pet of the month. However, your feminine instincts know the tactics necessary to divert his attention from this source. Yet, this totally logical device is a real Fast Sort.

Your conversation becomes limited to his ravings in Syntax Errors. Occasionally your intercourse is interjected by Dumps and Traces. Now and again, he wafts into graphic detail on Machine Language - does this mean it talks to him ? But when he wakes up in the middle of the night, proclaiming he must Poke and Peek, you distinctly feel he is being obscene.

Disrupted

My life was disrupted thus about 12 months ago. Before then I was blissfully ignorant of mainframes, minis and micros. I vaguely thought of them to be some new dress fashion. Now my Lord and Master has urged me to abandon all former ways. I have been instructed to discover the menacing Word Processor; fathom the meaning of Interfacing and Bus -which seems a rather ridiculous notion to me. No longer can I visualise Rams gaily prancing around fields on a bright Spring day. I hold chips I cannot devour (with or without vinegar) and foresee that ships are not the only entrants to ports. All these new innovations have been mind-boggling for a simple soul like myself, but the worst is yet to come.

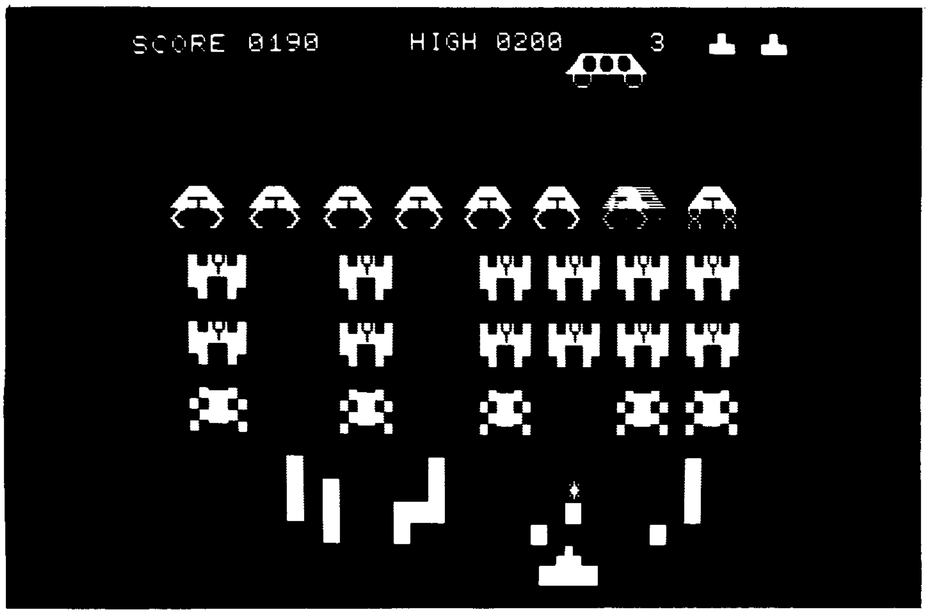
I have a fervent paranoic desire to personally hang, draw and quarter the man (and it could only be a man), who developed the Space Invaders program as Pet food. I spend hours watching the love of my life torn in agony as he tries to combat his wits against those infernal descending little blobs. His face alters rapidly while his body becomes rivetted to the machine. It reminds me of a Jekyll and Hyde film I once saw. Every muscle tenses as he attempts to bash hell out of the 'A' button whilst co-ordinating his movements on the '4' and '6' keys. He becomes impervious to all around him (especially me).

Bad Enough

You think this is bad enough - not so. Somewhere along the line, some bright enterprising little spark has manufactured a speaker unit for our dear little Pet. Now not only can you see the Invaders advancing but their speedy promenade is accompanied by weird sounds reminiscent of a Benjamin Britten opera. Like chalk screeching on a blackboard the excruciating 'Bleep-Bleeps' echo through your brain. This repetitious drumming is occasionally interspersed with a high pitched police siren type of noise announcing the arrival of a trolley bus. The bonus points for this hit are too much for your man to resist. He tracks it along, only to misfire and be bombed by an un-looked for Invader. Too much! Too much!

I am an optimistic person and believe that there is light at the end of every tunnel. There is an old saying - "if you can't beat them, join them". So the answer to this conundrum - we have two Pets, his and hers.

Close encounters of the fourth kind



Business User's Column

Barry Miles

This column will examine further the question of program and data security. You will recall that last time we began to examine the possible hazzards and their prevention and minimisation. Perhaps now would be a good time to emphasise that I do not have a negative attitude towards microcomputing, and indeed think it to have an extremely important part to play in increasing business efficiency. However, such enthusiasm should not be allowed to blind one to the danger inherent in handing over important activities to an alien machine, which may not be fully understood by the people operating it. It is a tool, to be used for easing your work. But, like all good tools, it needs care in its use. A sharp chisel, in the hands of a sensible user is a safe tool, because the pressure needed to operate it is far less than that needed to operate a blunt one. However, such a chisel, used by someone who does not know how to protect himself from harm, is dangerous indeed.

Major Dangers

The major dangers of employee dishonesty seem to fall naturally into two categories: Fraud, and data theft. Fraud takes two main forms in data processing: one consists of changing the program so that it operates in a way which distorts the data produced, so as to fraudulently change accounting information. For example, one could arrange for the payroll program to give more pay to a selected group of employees, and collect a share of the spoils from each of them. This might be done by changing the gross pay and distorting the control totals. How might we deal with this danger? Well, the simplest way is to arrange for the original program or security copy, kept out of the hands of the usual operator, to be run at irregular intervals by another operator, and the results checked against those produced in the ordinary way. You may choose to let it be known that such checks will be part of the normal work processes, "As the auditors insist", so that the deterrant effect may be achieved.

This type of fraud requires that the operator shall have sufficient knowledge of computer programming to be able to alter the program sucessfully. If the program is in machine code, this will be unlikely, and even simple alterations to someone else's program are not too easy to make, because there is always the possibility of the changes‘ having subtle unforseen effects on other parts of the program.

"Need to Know" Principle

Another way of protecting your programs from this type of tampering, is to ensure that they will run immediately on loading. This ensures that they cannot be listed in order to be changed. Not many commerical programs have this facility, but there are a few programming experts who could add it for you. Alternatively, it is now possible to buy a program which will change your programs in this fashion. Whilst you are at it, you may wish to commission work to be done to make access to your data-file impossible without use of a password, and to restrict knowledge of the password to those few people who must have the information in order to carry out their work: back to the "Need to know" principle!

Data Security

Data security is another matter. The extreme portability of your data disks is a matter for careful thought. The dangers include theft of information about your customers to enable a competitor to steal your business, theft of customer information for purposes of blackmail of customer or, if you are foolish enough to understate your income to the Inland Revenue, of yourself!

It is not only the theft of your own disks which must be prevented, but the information upon them must be safeguarded. This means that unauthorised copying of disks must be prevented. Duplicating a disk takes very few minutes, and the disks themselves can easily be carried into and out of your premises in any convenient briefcase or bag. So what can we do?

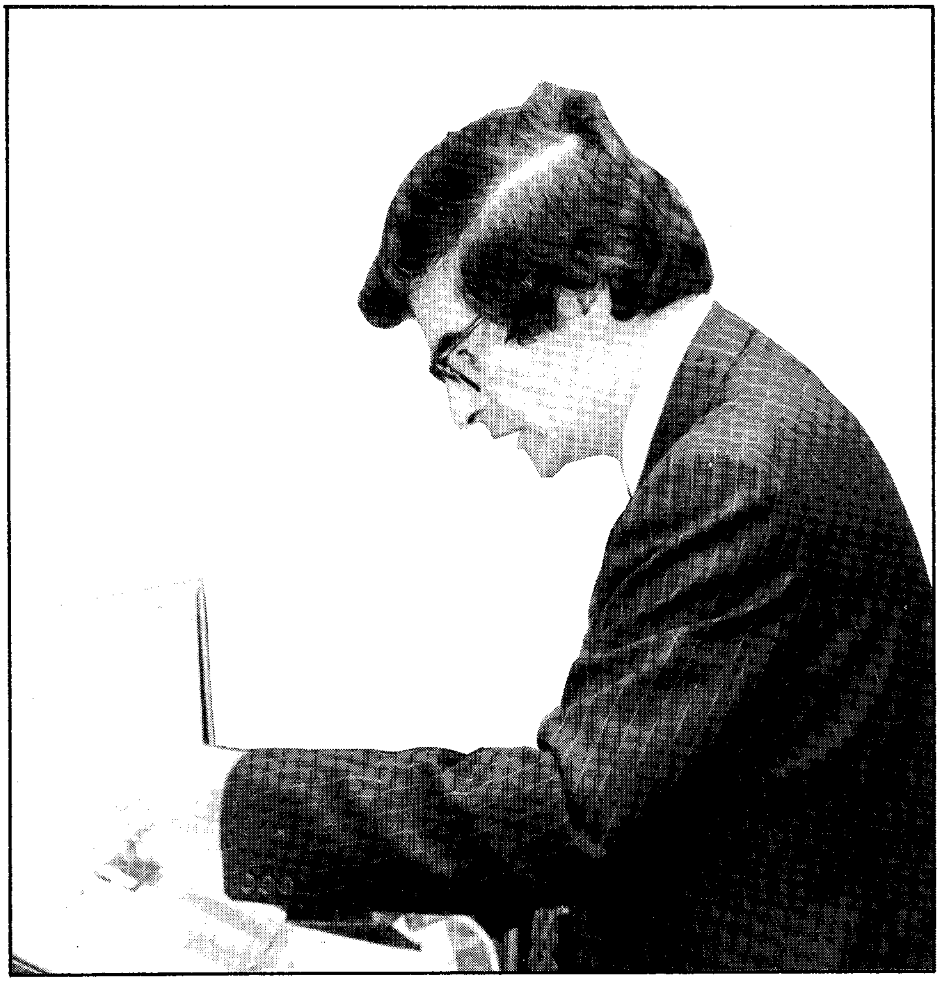
The main thing seems to be to avoid disks coming into the premises which are not the property of your business, **AND IDENTIFIABLE AS SUCH.** A cheap way of doing this is to have special disk labels printed, numbered, and in distinctive colours, and to ensure that all disk supplies are re-labeled as soon as they arrive. A record of receipts and issues of disks should be kept, with dates and recipients recorded, by some member of staff who has no other dealings with the computer installation. It should be clearly understood that alien disks will not be tolerated in the machine, and you should keep a wary eye open for infringment of this rule. If your staff become so keen on games that serious work is endangered, this will go some way towards reducing the difficulty.

It will be readily appreciated that many businesses will not be large enough to enable the various security methods suggested in this series to be carried out in full, but this will be largely compensated for by the fact that the owner of the smaller business can keep a closer eye on all the events in his business, than can the management of a larger organisation, which must rely on formal procedures to provide the necessary degree of protection.

You should not be reluctant to discuss the question of program and data security with your dealer, when considering buying a program; taking an interest in whether any precautions have been built into the program, and whether they can be custom-designed to your requirements. A useful enhancement would be to have the program stop operation at intervals, insisting that backup disks are recorded before continuing operation.

Consultation

A useful way of getting things organised in the most satisfactory manner is to arrange a consultation between your auditor and a skillful programmer, discussing all aspects of program and data security, and resulting in the programs being adapted in the most effective way, before being put into use. This will ensure that time is not wasted in setting up procedures which the auditor thinks are insecure, or in considering in detail suggestions from the auditor which the programmer can tell you immediately are not suitable for implementation, because of system limitations, of which neither you, nor the auditor are aware.



Barry Miles meaning business

Such a consultation will be even more important if you are going to the trouble and expense of having programs specially written for you.

Program Selection

Let us now examine the availability of programs which are so widely useful that any business would be justified in buying them, without bothering to go into them in the fashion which I have suggested in previous articles in CPUCN.

What we are looking for is routine operations of such universal applicability that it is almost inconceivable that they would not be of use in almost every business which is likely to acquire a Commodore system.

Firstly we should consider what we are aiming to do. It is usually the case that the machine's purchase will have been for a particular purpose, and the purchase will have been justified on the basis of time saved or increases in efficiency or speed. We are now interested in obtaining good value for money by using spare machine-time in a cost-effective fashion. This implies being able to use programs intermittantly, fitting the operations around the existing machine-load schedule.

Word Processing

An obvious choice is word processing. In case any reader is not fully aware of what this means, I will just say that it is a form of highly intelligent typewriting, enabling an amazing range of editing, alteration, storage, and printing facilities to be used.

We are fortunate that very fine machine-code word processing programs exist for the Commodore machines, and they offer facilities matching those on dedicated word processing machines costing many times more than the equipment we are using.

The selection between the two programs which lead the field is a matter of personal choice, some prefer *Wordpro*, and some *Wordcraft*. If you are using the 40 column machine, then using Wordcraft is the only way to see on the screen how your formatted output will look. If you are using an 8032 then you have the choice between Wordpro 4 and Wordcraft 80. Either of these programs will suit you very well, and many an installation is used just for word processing and is fully justified by so doing. With a daisy-wheel printer you can impress your clients or customers with the skill of your typists, who appear to reject anything less than perfect output. Some people feel insulted by receiving a letter printed by computer, so a daisy-wheel printer will save their feelings, and maintain good customer relations.

I suggest you read the reviews of these programs in earlier editions of the Commodore Newsletter, and make up your mind from them. If your requirements are not so sophisticated, you may find Papermate an attractive, and cheaper alternative.

Visicalc

Another program, this time of astonishing versatility is Viscalc. Some people have the mistaken idea that this is a financial modelling package, and as such, only suitable for use by accountants. This is not true. It is really a way of carrying out tabulations of any figures, and anyone who is called upon to set figures out in rows and columns for any purpose, and who needs to be able to carry out calculations for the purpose, or change any figures, should buy this program without delay. In my opinion, used properly, it is capable of being the most exciting innovation for microcomputers since the word-processing programs were developed. Unfortunately it suffers from some defects which limit its usefulness, indeed, I have been forced to write some programs myself to eliminate two of its most serious defects: firstly to enable printing to be carried out on a non-Commodore printer, and secondly, to enable the formulae which you are commanding Visicalc to implement, to be printed. Without this facility you are in the position of a programmer who is not able to check the logical flow of the program which he has written, because he cannot list it!

However, even with these reservations, I cannot recommend the program too strongly. The latest versions print on any printer, having been modified by Harry Broomhall for the purpose.

If you are interested in carrying out some Basic programming yourself, or expect one of your staff to do some programming, then Basic Aid is an amazingly cheap program from Commodore, which will be enormously helpful. The more ambitious will go for some of the attractive add-ons.

Reviews

Cathedral saved from collapse by Constructional Software

Well that's not quite the case but the editor suggested this might be an eye-catching little article about some unusual application.

Constructional Software programs are generally for use, as the name implies, in the construction industry. They are not usually spectacular in effect — not for us the claims of revolution in small businesses. At least, it depends what you mean by revolution. Ours is a quieter revolution.

Take an example: frame analysis is a job regularly done by structural engineers. Before the computer it was done the long way using a slide rule and log tables. A bit later bureau service was available using computer techniques, then a terminal might be seen in the design office using a telephone link to a large computer. While this was happening the powerful programmable calculator appeared and calculations were lovingly entered for immediate results. This adds up to an established background of computer use. Constructional Software offers, in its Keypac and new Strider suite, programs which enable the CBM microcomputer to combine the roles of powerful calculator and terminal link. Delays and frustrations experienced with either postal bureaux or time sharing terminals are gone.

Set this sort of function beside a general accounting package, a Wordcraft 80 and Ozz and then, perhaps, we have a real revolution.

These comments apply to most of our products: after all, no one is going to get excited about Keypac 20 for drainage design. Critical path analysis is seen by many as a tedious and unending calculation because everything keeps changing. Use Prenet for your small networks or the new Project program (shortly to be released) for large networks and updating becomes a real pleasure. We have one customer who freely admits that he only wanted Prenet to impress his customers — when he had used it for a couple of weeks he 'phoned to say that he used it all the time and wouldn't be without it.

Workmanlike Programs

I hope by now that you can begin to see a picture of a series of workmanlike programs solving established problems. No bells and whistles but reliable tools. Just as the engineer might have kept a slide rule in his pocket because it seemed to answer so many questions then a PET on your desk could feel the same.

On the other hand, integration is a common topic when computers and their programs are being discussed. Again, it depends what you mean by integration. If you mean a series of programs which will all talk to each other and are linked into an all-singing all-dancing megalith, then forget it — that sort of common language is about as useful as Esperanto (with respect). Everything to everyone and nothing to anyone. See yourself as integrator: a central control with access to the appropriate tools. Our Specwriter package is a good example. Take a regular (and astonishingly good) word processor package in Wordcraft 80 and couple it with our massive National Building Specification text database and, between the two, you have something which will not only write a specification but also send out your Christmas cards, write your letters, type your Section 27 notices and prepare articles like this. A good example of two useful tools combined to satisfy both general and specific applications.

Don't be put off by trying to find something that does everything and getting hopelessly confused in the process. Let the PET micro do something, anything, which takes advantage of its power and accessibility and you will find that the computer is friendly and useful and gives you more time to worry, or not as the case may be, about the things worth worrying about.

Roy Stephenson — Claremont Controls Ltd. A Commodore Approved Product Supplier.

Great Traditions taken into the Micro-Age

The Commodore PET with a Landsoft program has chalked up two firsts in the world of fine food and wine in the midst of the yachting mecca of Lymington. The Stanwell House Hotel and Railings Restaurant has been able to take the traditions of comfort, service, food and great wines into the microcomputer age; and was the first establishment to take the Landsoft G.B. 2 System on a Commodore PET.

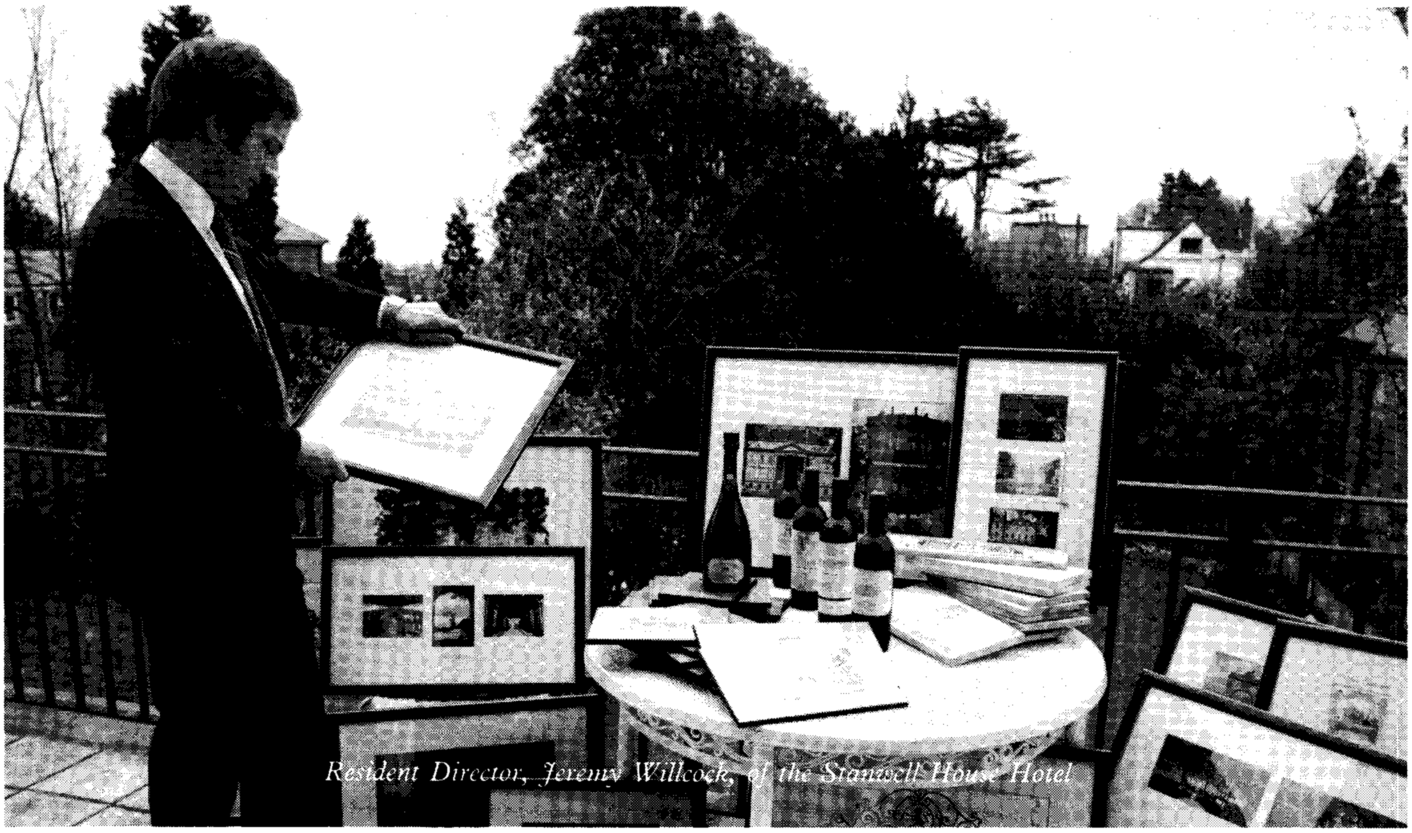
This both bills and analyses all charges automatically, enters standing charges for the small items that make a major contribution to the well being of guests. Early morning teas, and fresh juices, late night sandwiches or newspapers are all automatically entered. But this Landsoft PET system is not only used to incorporate numerical amounts.

The Stanwell House Hotel was the first to use the system for bedrooms that are named, not numbered. All the bedrooms are called after famous wines and the system copes with the names — and more. For, guests ordering their wine after which their bedroom is named are offered a discount. And the system takes care of that as well.

Says Jeremy Willcock, Resident Director, "The system is ideal for individual high class hotels. I cannot understand why the majority — which after all are owner-run like ourselves — have not installed a micro-aid which is alpha numeric."

The PET system in the Stanwell House Hotel handles guest billing and payroll using Landsoft programs, and wordprocessing using Professional Software's Wordpro 3 through which it can take care of menus and wine lists, etc.

This facility is particularly important to the restaurant as the demand for fine wines is growing all the time and different wines and vintages are being introduced constantly. These can be added very simply with the Wordpro 3, providing the restaurant with a list which is always



Resident Director, Jeremy Willcock, of the Stanwell House Hotel

up to date and clean with nothing scratched out or listed but out of stock.

"We ran the system as a back-up for a month while we trained ourselves in", says Jeremy Willcock. "It became our primary system on January 1st 1981. Since then we've had no problems — except for the time when a new receptionist put her fingers on a disk!"

Landsoft is a Commodore Approved Product Supplier, and their telephone number for further information is: 01-339 2476.

Master Directory/Index *Supersoft*

This program is a "must" for anyone who has a large number of disks, and uses a 4040 or 3040 Disk unit.

Written partly in machine code for speed, it takes the directory from each disk, sorts it into ASCII order, and stores it on the directory disk. This process is so quick that you can feed the disks into the unit as fast as possible; by the time you have hit the necessary keys, the drive light has gone out, and you can remove that disk while the sort takes place. Furthermore, you can update the details of any changes as a result of adding or scratching files, simply by re-entering the disk into the catalogue.

Being menu-driven the program is pleasant to operate, and you can list any selected disk's directory, find any program, (using just the first letter if you wish), delete an entry, find the number of blocks free on each disk, **or sort through the disk to find one with enough space on it for that long file you wish to create.** This last feature is very useful to help you get the maximum usage out of disk space. The program will also handle the master disk itself, and add it to the catalogue!

Criticisms? Well, I would like to see an 8050 version as soon as posible! Also, it would be better if the files were listed in columns, with the full screen used.

"Master Index"

The companion program "Master Index", will enable you to print out an alphabetical list of every program in the catalogue, at as high a speed as your printer is capable. You may feel that this is a bit expensive at £12, but that will depend on the number of diskettes you have, and whether time is money to you!

Supersoft also offer a machine code "Compactor". There was a Compactor program in BASIC, published by "Compute" magazine, but, although it is reliable, it is rather slow, and this program represents a quantum speed improvement.

What it does is to remove all the REMs in a program, all the spaces, and semi-colons which are unnecessary. Then, preserving all lines which are targets of GOSUB or GOTO, or RUN, it links lines together, to form lines of up to 250 bytes long!

People who are going for maximum space-saving and speed, will find that examination of the "compacted" program will show ways of changing the code, reducing the number of target lines, so that compacting the program again is very fast to use, and is available for both New ROMs and BASIC 4 machines. It represents the next best thing to a compiler, and its speed is such that it is perfectly feasible to use it at runtime only, so that you do not have to store two versions of the program if you don't want to. The instructions show you how to protect the routine, very simply, so that you can run a series of programs, compacting each one as you use it. At £15, this represents very good value for money.

There are a large number of programming-aids available now, and it is difficult for the user to decide between them. They vary from out-and-out aids to the actual process of programming, e.g. Toolkit, Programmer's Friend, through those which are both aids to programming and routines for incorporation in programs, e.g. Commando and Disk-o-Pro, to those which are intended for incorporation within programs only, e.g. Multisort.

PET Enhancement Package

This package, available from Supersoft again, is mainly a set of routines for incorporation into your programs, although it does have features for use in direct mode also.

The program is not on a chip, and if you want mobility between machines, this is important. With true portability in mind, the author, Mike Todd, has arranged that it is possible to save the P.E.P. tacked onto your program, by using a normal-looking BASIC SAVE command, and a LOAD will bring the enhanced program into your machine.

It offers a computed GOTO and GOSUB; a POS function, which tells you where a substring occurs within a string; a user definable print routine, the ability to turn the package on and off, from software (this is important, because the use of the interrupt routines implies a speed penalty, and there is no point in suffering this if speed is paramount, and the section of code being processed does not call on any P.E.P. routines.)

Also available is a repeat key, (I should add that the program is for New Roms only, a BASIC 4 version not yet being available); the stop key can be disabled and enabled at will; the upper and lower limit of RAM can be set; double density plotting is available ("Little squares"!); variables can be zeroed en bloc, degrees can be converted to radians, and vice-versa; the case of alpha characters can be inverted in a string, this will help you convert programs written on the BASIC ROMS to run with ease on the New Roms.

A very powerful command is the "DO" Command. This is in the form of "DO A$", where A$ contains a Basic statement. This will have many uses for the lively-minded programmer, offering as it does the possibility of the equivalent of self modifying programs.

Also included is a carefully thought-out set of input routines, to help you prevent the user who hits return with no data from dropping out of the program. This uses a special variable to tell you what has happened, so that the program can branch as you wish.

All in all a very interesting package, and cheap at £25.

For Computer Aided Instruction:— MTC PILOT

An interpreter for an extended version of common PILOT, implemented as a machine-code program for any model of 16K or 32K PET computer, using cassette or floppy disk drive.

Common PILOT is a powerful, dedicated language for writing interactive instructional programs, developed at Western Washington University. Using this language, it is possible for novices to start writing useful instructional programs within four hours of first sitting down at a computer keyboard. Thus common PILOT is ideal for many applications in schools, colleges and universities. Further applications for easily-written interactive programs arise in business and industrial training.

The language will also be found useful in Computer Science teaching. The simplicity of its structure and its ease of use make it a very suitable first language for a beginner. Advanced programmers will find that common PILOT has some very powerful features, including easy program self-modification using the 'execute indirect' command.

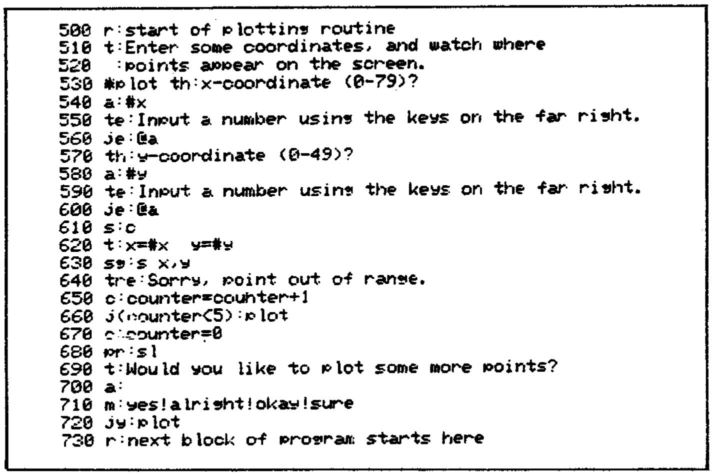
MTC PILOT is an adaptation and extension of common PILOT, implemented as a new interpreter for the PET. All the features of common PILOT are supported, including automatic editing of user input, sophisticated 'window strings' response matching, 'execute indirect' command, random block access, floppy disk file handling, and 'escape' and 'goto' options allowing run-time modification of program execution. A number of extra features have been added, including extended PET graphics, sound generation, full BASIC string handling and string arrays, plus PEEK, POKE and USR facilities. Using MTC PILOT, programs can be entered, edited, saved, verified, loaded and run just like BASIC programs. The full PET screen editor can be used with PILOT programs plus extra facilities like automatic line numbering, renumbering, block deletion, listing to printer, and repeat on all keys.

By making full use of routines already available in PET ROMs, all this has been achieved in approximately 5K of machine code. This code relocates the top of RAM when PILOT is loaded, leaving the rest of memory free for PILOT programs.

Although MTC PILOT will function using cassettes only, users of Commodore disk drives and Mu-PET will find a number of special features in disk versions of the program. An enhanced DOS SUPPORT is incorporated, which checks the error channel after every disk access, so that any errors are trapped automatically. In addition to this, the random block access file facilities allow student records to be kept so that progress can be monitored, and also allow a single PILOT program to access well over 100 kilobytes of text.

MTC PILOT - A Sample Routine

From a program designed to introduce Cartesian coordinates:—



```
500 r:start of plotting routine
510 t:Enter some coordinates, and watch where
520  :points appear on the screen.
530 *plot th:x-coordinate (0-79)?
540 a:#x
550 te:Input a number using the keys on the far right.
560 je:@a
570 th:y-coordinate (0-49)?
580 a:#y
590 te:Input a number using the keys on the far right.
600 je:@a
610 s:c
620 t:x=#x  y=#y
630 sg:s x,y
640 tre:Sorry, point out of range.
650 c:counter=counter+1
660 j(counter<5):plot
670 c:counter=0
680 pr:s1
690 t:Would you like to plot some more points?
700 a:
710 m:yes!alright!okay!sure
720 jy:plot
730 r:next block of program starts here
```

The action of this program is as follows:—

500 remark — ignored on execution
510 -520 type instructions
530 starts with a label which can be used as the destination of branch instructions. Followed by a type instruction with a 'hang' modifier, which inhibits carriage return/line feed.

540 -560 accepts an input from the user, and stores the first number input in variable x. If no number is input an error message is printed, followed by a jump back to the accept command.

570 -600 accepts the value of variable y in the same way.

610 -640 clears the screen, types the chosen coordinates, and plots the point by setting point (x,y) in a double-density graphics display. If an error in encountered due to the point being out of range, an appropriate message is displayed in reverse field.

650 -660 increments a counter in a compute statement, then if the counter is less than five performs a conditional jump back to label *plot.

670 -680 resets the counter, then in a problem command sets automatic editing options to remove all spaces from user input, and force it into lower case.

690 -700 types a question, then accepts an input from the user, which is automatically edited and stored in a special 'answer buffer'.

710 searches the answer buffer, looking for a match for any of the strings specified anywhere in the buffer.

720 -730 if the match was successful, branches back to the label *plot. Otherwise execution continues from line 730.

MTC PILOT — Programming Features

- Allows simple linearly-structured programs, with remark statements for self-documentation.
- Type statement with hang and reverse-field modifiers prints text, variable values, and full PET graphics.
- Special screen command gives programmed cursor control and screen poking, plus computed cursor positioning and double-density graph plotting set reset and test commands.
- Crash-proof accept command allows input into answer buffer, or into numeric or string variables. Single modifier can be used to get single character, and exact modifier to suppress automatic editing.
- Problem command allows various automatic editing options to be set, including removing spaces or forcing upper or lower case.
- Versatile match instruction can search for a number of alternative responses anywhere in the answer buffer. Special characters match with any string or with any single character, allowing compensation for simple spelling errors.
- Jumps can be made to labels of up to six characters anywhere in the program, or to the last label, last accept, next match, or next problem command. Jump modifier on match gives automatic jump to next match if current match fails.
- Subroutines nestable up to 24 deep. Subroutine end instruction can be used with a label.
- Computer command gives all BASIC functions including PEEK and USR. Special string editing commands can force upper case, force lower case, capitalise, remove or replace specified characters.
- Numeric or string arrays with any number of dimensions as in BASIC.
- Execute indirect command allows strings to be concatenated, and the result to be executed as a line of PILOT. Allows easy program self-modification.

- Random block access file commands usable with Commodore disk drives. Particularly useful for keeping student records, and/or for storing extended passages of text.
- Sound effects command gives music or noise generation on a simple external sound box.
- Escape and goto options allow user to jump to a label or call a subroutine in the middle of normal program execution. Can be used to provide a calculator facility in scientific exercises.
- Six types of conditional execution, including on match flag, error flag or relational expression.

MTC Pilot - Editing Features

- Full PET dynamic screen edit. NEW, LOAD, SAVE, VERIFY and LIST function as in BASIC.
- Programs can be run from first line or from a specified label. Also possible to goto a specified label.
- ? can be used to check values of variables during debugging.
- Single command gives listing on Commodore printer in either upper or lower case modes.
- Automatic line numbering in steps of any size.
- 3-parameter line renumbering facility.
- Block deletion facility.
- Direct mode commands BASIC and PILOT allow transfer between languages.
- Repeat function can be enabled on all keys (also functions in BASIC)
- Upper and lower case commands give easy transfer between alternative character sets (also function in BASIC).
- Disk version incorporates enhanced DOS SUPPORT, with full error trapping on all disk access, including program load.

The Registered Users Scheme

All purchasers of MTC PILOT are automatically entered in the registered users scheme, entitling them to the following:—

- MITAC Publishing will supply extra copies of the program and manual to registered users, for only 20% of the original cost.
- Later updates of MTC PILOT will be made available to registered users at substantially reduced prices.
- MITAC will circulate information about programs written in MTC PILOT as they become available. This includes programs published by MITAC and approved programs released by other companies.
- Modern Tutorial College will endeavour to provide support to help with any difficulties encountered in the use of MTC PILOT.

MTC PILOT is available in a package consisting of the interpreter, a manual, and sample programs on tape or on disk, for £65 plus VAT. Enquiries to MITAC Publishing Ltd., Modern Tutorial College, Kilburn Lane, London W10 4AA. Tel: 01-960 5899.

Basic Programming

8032 — Screen Facilities

There have been a number of requests for more information about the 8032 screen control facilities, basically how to use them. These are a nice feature of the machine, and set it apart from earlier Commodore computers. Since these facilities are not explained in the manual that accompanies the 8032 they are explained below, along with one or two other little-known facts. All of the commands can be used quite easily from within a Basic program by a series of CHR$ commands. If you're new to PETs, and CHR$ (otherwise known as character string) doesn't mean too much to you, here goes.

Turn the PET on, and type printchr$(65) and hit the return key. The result will be a letter 'a' appearing on the screen. If you type printchr$(65+128) you'll get a capital 'A' this time. And so you can go on through the alphabet, and onto all the graphics characters and the various numbers and symbols that appear on the keyboard. Now, some of the chr$ numbers do not produce characters on the screen, but perform actions. For instance, if you type printchr$(147) the screen is cleared. Some of the more interesting chr$ numbers are shown below, along with an eplanation of what will happen. Experiment yourself — there's no chr$ command that will do any damage to your PET!

Number **Action**

7 Sounds the bell.

14 Sets the PET into lower case mode. It is in this mode when you turn it on.

21 Moves everything below the cursor on the screen up one row.

25 Moves all the screen up one row.

142 Sets the PET to upper case mode — this gives you all the interesting characters that you can't see on the keyboard.

149 Moves everything below the cursor on the screen down one row.

153 Moves all the screen down one row.

Onto the screen control features. There are two CHR$ numbers that set respectively the bottom right hand corner and top left hand corner of a 'window' on the screen. This would then happily allow you to show changing information in that window while the rest of the contents of the screen remain unchanged. This has many obvious uses, for instance in education where the ability to have a display on the rest of the screen, and the instructions for proceeding within the window, would be very useful. To set the bottom right hand corner, position the cursor where you want that corner to be, and type printchr$(143). Do the same for the top left, only this time use 15 instead of 143. Experiment!

An alternative to using CHR$ is to use the POKE command. This gives you rather more control over the setting up of the window. There are four necessary POKEs to do, and these are:- To set the top of the window — type POKE224,w
To set the bottom of the window — type POKE 225,x
To set the left margin — type POKE226,y
To set the right margin — type POKE213,z

I'll leave it up to you to play with values for w, x, y and z, but they are within the ranges of 0 to 25 for w and x and 0 to 80 for y and z. Once you've created your window and had a play, press HOME twice to escape from it. As I've said, experiment with the commands and see what happens — you can't do any harm. There are a number of POKE commands that do interesting things. Typing POKE231,a where a is any value between 0 and 255, is quite good. It alters the speed of the little bell that sounds when you're reaching the right hand margin. 0 turns it off altogether, and 255 sounds for a long, long time. To make the bell sound without having to go to the right hand margin, type printCHR$(7) followed by return. Another location with an interesting use is 144 — POKE144,88 disables the stop key, and POKE144,85 re-enables it again.

Onto setting tabs. You've no doubt noticed that there is a TAB key on the keyboard, but are not quite sure how it works or even what it does. To set a tab in column 8, type POKE1007,1 — for column 16 POKE1008,1 — and so on in jumps of 8 up to column 80 and POKE 1015,1. Pressing the TAB key will then move the cursor to wherever the next tab is set. To remove the tabs, poke the same location but with 0 instead of 1.

Hopefully you now have a clearer idea about at least some of the workings of the 8032, and can begin to work on your own programs.

Use of Arrays in BASIC: An Introductory Guide

Margaret Skinner, Program Advisor,
Computer Centre,
King's College London

An array is a set of consecutive locations in the store of the machine which is given a name conforming to the usual rules for the formation of variable names, i.e.
(i) The name consists of one or two characters
(ii) The first character must be a letter
(iii) The second character must be a letter or a digit
(iv) If the array is to hold character strings then a dollar sign is appended to the name.

Thus A, K9 and OK$ are all legal array names, but MI5 and 3A are not.

Individual elements of an array are identified by the use of subscripts — the elements of array K are K(0), K(1), K(2), K(3) ... and so on.

Use of the DIM statement for dimensioning arrays

The DIM statement is used to reserve a specified amount of room in store for each array, and every array that is to consist of more than eleven elements must be referred to in a DIM statement before it is referred to in any other statement. Smaller arrays may appear in a DIM statement and for reasons of clarity it is good practice to dimension all arrays in this way, whatever their size. The following statements are examples of valid DIM statements:

```
DIM A(20)
DIM P(50), Q(20), R$(10)
```

A program may contain several DIM statements, but no array should be dimensioned more than once. Although it is possible for these statements to appear scattered throughout the program it is usually clearer and tidier to group them together at the beginning of the program.

There are occasions, however, when considerations of the speed of execution of a program require that the DIM statement should appear later.

An array may be dimensioned dynamically during program execution; thus the statement

```
DIM K(N)
```

reserves N + 1 locations for array K (remember that the first location is K(0)). The value of N must be set in the program before the DIM statement appears.

Using an Array

Suppose a factory's quality controller wishes to examine a batch of widgets. All widgets whose length deviates by more than 10% from the mean length are to be discarded, and so she needs to calculate the mean length and then examine each widget in turn to see if it is to be discarded. We assume that the data is to be entered from the keyboard. The following program will allow the data to be stored in array L and then processed.

```
 10  INPUT  "HOW MANY WIDGETS IN THIS BATCH ";N
 20  DIM L(N)
 30  SM = 0
 40  PRINT "ENTER ONE LENGTH ON EACH LINE AND PRESS RETURN"
 50  FOR K = 1 TO N
 60  INPUT L(K)
 70  SM = SM + L(K)
 80  NEXT
 90  REM CALCULATE MEAN AND 10% OF MEAN
100  MN = SM/N : TN = MN/10 : PRINT "MEAN IS ";MN
110  FOR K = 1 TO N
120  IF ABS (L(K) - MN) > TN THEN PRINT "WIDGET";K;
     "LENGTH ";L(K)
130  NEXT
140  END
```

Note that as the data is input a running total is kept in location SM (Line 70). SM is initially set to zero in Line 30; strictly speaking this line is not necessary on the PET as all variables are initially set to zero automatically, but as this is not true of all machines you are advised to get into the habit of initialising variables yourself. The mean and 10% of the mean can then be calculated easily (Line 100). The ABS function (Line 120) is a standard function used to find the absolute (i.e. disregarding sign) value of a number. When this program is RUN the quality controller will be able to see both the serial numbers and the lengths of the widgets to be discarded.

Arrays of more than one dimension

So far all reference has been to arrays with just one dimension but it is possible to use any number of dimensions. In practice one-dimensional arrays are used a great deal, two-dimensional arrays are used occasionally but higher dimensions usually create more problems than they solve. Users familiar with matrices will be used to the idea of "rows and columns" and a two-dimensional array can be used to store data in this form. Let's suppose that 100 students each take 3 examination papers; their marks could conveniently be stored in a 100 by 3 array. Thus we might have

```
10  DIM MK(100,3)
20  FOR K = 1 TO 100
30  INPUT MK(K,1),MK(K,2),MK(K,3)
40  NEXT
```

Each student's set of marks will be entered on one line, and will be stored in the appropriate place in the array.

The DIM statement actually reserves 404 elements of store, since the first subscript can range from 0 to 100 and the second from 0 to 3. If space is at a premium then the following would be more efficient:

```
10  DIM MK(99,2)
20  FOR K = 0 TO 99
30  INPUT MK(K,0),MK(K,1),MK(K,2)
40  NEXT
```

If, however, clarity is more important than conservation of space, then the original form is to be preferred.

Conclusion

Sensible use of arrays makes programs easier to write and easier to debug. Their most important use is probably for the storage of data which is to be processed in various ways as illustrated in the examples in this article, but the experienced programmer finds many other situations in which judicious use of an array will make her program more elegant.

Hurray for Arrays

Following on from that article on arrays, we have an interesting letter from Dr. B. Orchard of the Horticultural Advisory Service in Guernsey. He writes:—

In Volume 2 issue 5 you give two programs as examples of methods of redimensioning arrays.

These do not work because new simple variables are defined after the top-of-arrays value has been stored. This is not mentioned directly in the article and I am surprised that a fundamental point of this nature has been overlooked. (Ed. sorry ....).

A simpler method seems to be to measure the size of the temporary arrays and then re-set the top-of-arrays pointer by this amount immediately before they are re-dimensioned. There is then no risk that using a new variable within the program will suddenly stop the re-dimensioning system working. An example is enclosed.

The example.

```
115 DEFFNP(I)=PEEK(I)+256*PEEK(I+1):DEFFNH(P)=INT(P/256)
12Ø DEFFNL(Q)=Q-256*FNH(Q)

15ØØ REM DIMENSION TEMPORARY ARRAYS
151Ø P%=FNP(46)+DP%:POKE46,FNL(P%):POKE47,FNH(P%)
152Ø DIMD(G+2),M(V),N(V),N$(V),F%(V),G(V,V),F(V,G)
153Ø DP%=P%-FNP(46):RETURN
```

V here is just a variable number, whatever you want to dimension the arrays. All well and good, but the saga doesn't stop here. I showed Dr. Orchard's letter and sub-programs to Mike Gross-Niklaus, Commodore's Software Manager, who came up with the following program which deals with the problem in a rather more straightforward manner. As is Mike's good habit, the program is heavily REMmed for ease of understanding. Incidentally, it helps myself and readers of the magazine if ALL submitted programs come in like this — REMs can always be removed after the program has been digested and understood. The program:—

```
1Ø PRINTFRE(Ø):REM SPACE BEFORE BUSINESS COMMENCES
2Ø DIM A$(1ØØ):REM TYPICAL ARRAY
3Ø PRINTFRE(Ø):REM SPACE REMAINING
4Ø POKE46,PEEK(44):REM MOVE START OF BASIC ARRAYS
5Ø POKE47,PEEK(45):REM TO END OF BASIC ARRAYS
6Ø DIMA$(1ØØ):REM DOES IT WORK
7Ø PRINTFRE(Ø):REM YES IT DOES, NO EXTRA SPACE USED
```

Thanks to Dr. Orchard and Mike Gross-Niklaus.

Auto Data Entry

Trevor Lusty, who wrote the review of Pascal in the last issue, has come up with the following program:-

This program has been used to enter large amounts of data without entering line numbers or 'DATA'. The following instructions assume that the user does not have a Toolkit fitted.

1) LOAD the 'auto-data' program into the Pet.
2) LIST the 'auto-data' program onto the screen.
3) LOAD the program to which DATA is to be added.
4) HOME the cursor and press 'return' to append each line of the 'auto-data' program.
5) RUN
6) Enter a line of data and press 'return'.
7) Repeat step 6 until all data is entered.
8) Press the STOP key to exit.
9) Delete the 'auto-data' program.

The program itself is as follows :-

```
1 REM *** AUTO DATA ENTRY ***
2 REM ***  TREVOR LUSTY.   ***
3 INPUT "[clr][rvs]START AT LINE ";N
4 INPUT "[rvs]LINE INCREMENT ";K
5 IFN<16 THEN N=16
6 PRINT "[clr][crsr down][crsr down]";N;"DATA ";
7 GETA$:IFA$=""THEN7
8 IFA$=CHR$(13)THEN1Ø
9 PRINTA$;:GOTO7
1Ø PRINT:PRINT "N=";N+K;":K=";K
11 POKE158,8:POKE623,13
12 POKE624,13:POKE625,71
13 POKE626,79:POKE627,84
14 POKE628,79:POKE629,54
15 POKE63Ø,13:PRINT "[home]";:END
```

Just remember to take care of your line numbers!

Inputs on INPUT

A comment from Donald Skene of Maidstone in Kent, on INPUT on the Pet. He remarks :-

I have come across a rather annoying problem with the Pet Input command which I have not seen reported elsewhere. Keying in a string should only cause the Keyed characters to be placed in the string variable. This may not be the case, however, if :

1. Any other I/0 files are open, and
2. The cursor 'wraps around' and appears in the next screen line.

If these conditions occur, then the prompt associated with the Input command is liable to appear as a part of the string variable. Thus, to be sure of accurate input, you must :

```
OPEN 1,15: ...

INPUT "PROMPT";I$: PRINTI$

IF LEFT$(I$,8) = "PROMPT? " THEN I$ = MID$(I$,9)

IF LEFT$(I$,1) = CHR$(34) THEN I$ = MID$(I$,2)..
```

It is not sufficent to test the length of the string because if you go over the screen wrap and then delete back again, you can have a string of less than 40 characters and still have the prompt included.

Note that you must test for a quote mark and that the problem becomes insoluble if some of the prompt characters are deleted during input. The term prompt includes the ? output with 'INPUT I$'.

The effect is rather unpredictable because it does not always happen if a program is run after inputting, saving or loading it. If it does not happen than it seems to be provoked by stopping the program with a null string and then running it again.

During the summer of 1981, Commodore will be moving to larger premises on the Slough Trading Estate. This is to cope with expected expansion over the next five years. Full details will be published as and when available, but until this is done the address remains as 818, Leigh Road, Slough, Berks.

M/Code Sequential Read Program

(From the West Australian Commodore Computer Users Association, Suite 3, 870 Beaufort Street, Inglewood, West Australia, 6052.)

I was prompted in writing this program by the frustration of trying to debug two other programs that purported to do the same thing. They were from Petsoft's 'A Hitchikers Guide to the PET' and from 'Library of PET Subroutines' by Computabits. Information, programs and program utilites are essential for the continued interest in Commodore Computers in all fields. It is a pity that so many programs and books are not fully debugged before distribution as this creates an air of unreliability that reflects on the product itself. It also causes tremendous frustration whilst programming at 1 o'clock in the morning !!!

Both the Assembler listings and hex code are enclosed for those who may not have an Assembler. If the routine is loaded after 'Dos Support', and the top of memory lowered to protect it, then they will co-exist very well. During programming it is very useful to be able to read a sequential file 'directly' without having to save the program currently in memory and writing a routine to read the sequential file. For speed reasons I did not use the Rom Screen routine as it is too slow, but instead utilised the Rom Scroll routine for effective display. To execute the routine enter SYS 32177, and then enter the file name. During display, the listing can be halted by pressing the stop key, and released with any other key.

You will note that the program initially sets up the fast screen poke; as this is not suitable for some machines it should be deleted in those cases.

By Roger Davis.
Chairman - W.A.C.C.U.A.

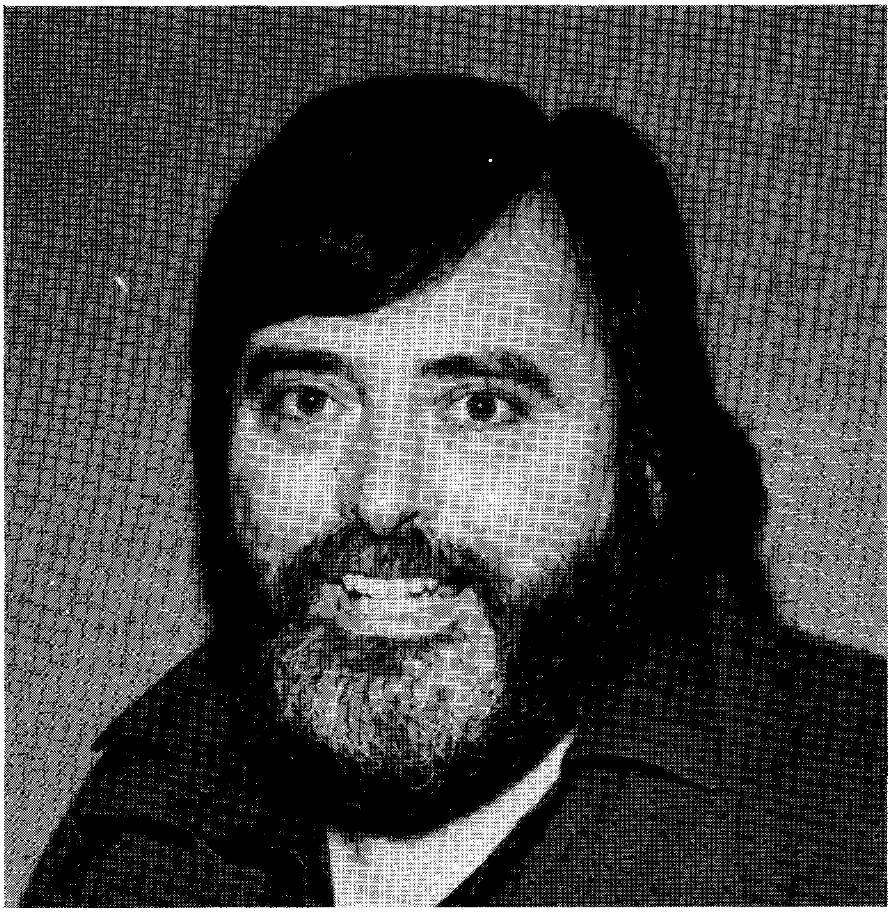
```
B*
     PC  IRQ  SR AC XR YR SP
.;  0401 E62E 32 04 5E 00 F4
.
.:  7DB0 AA A9 0E 8D 4C E8 A9 3E
.:  7DB8 8D 62 E8 20 6F C4 A9 00
.:  7DC0 85 DA A9 02 85 DB A9 02
.:  7DC8 85 D2 A9 08 85 D4 A9 02
.:  7DD0 85 D3 A2 00 BD 00 02 F0
.:  7DD8 07 E8 20 01 F3 4C D4 7D
.:  7DE0 86 D1 20 24 F5 A2 02 20
.:  7DE8 70 F7 A2 00 A0 00 8E 00
.:  7DF0 10 20 01 F3 F0 03 4C 03
.:  7DF8 7E 20 E4 FF F0 FB A2 02
.:  7E00 20 70 F7 A9 00 20 8C F1
.:  7E08 A4 96 D0 08 AE 00 10 A0
.:  7E10 00 4C 23 7E A2 00 A0 00
.:  7E18 A5 D2 20 AE F2 20 CC FF
.:  7E20 20 8B C3 A8 C0 0D D0 08
.:  7E28 20 3F E5 A2 00 4C EE 7D
.:  7E30 98 9D C0 83 E8 E0 28 90
.:  7E38 B5 4C 28 7E AA AA AA AA
.?
.
READY.
```

```
SEQSRC......PAGE 0001

LINE# LOC   CODE         LINE


0001  0000                                   ;*******************************
0002  0000                                   ;*
0003  0000                                   ;*        PROGRAM TO READ
0004  0000                                   ;*
0005  0000                                   ;*      SEQUENTIAL DISK FILES
0006  0000                                   ;*
0007  0000                                   ;*        VIA M/CODE ROUTINE
0008  0000                                   ;*
0009  0000                                   ;*******************************
0010  0000                                   ;*
0011  0000                                   ;*******************************
0012  0000                                   ;*
0013  0000                                   ;*
0014  0000                                   ;*        BY ROGER DAVIS
0015  0000                                   ;*
0016  0000                                   ;*     15TH - 19TH AUGUST 1980
0017  0000                                   ;*
0018  0000                                   ;*
0019  0000                                   ;*******************************
0020  0000                                   ;
0021  0000                                   ;
0022  0000                                   ; SYSTEM VARIABLES
0023  0000                                   ;
0024  0000                                   ;
0025  0000              LONAM=$DA             ;ADDRESS OF FILE NAME LOW BYTE
0026  0000              HINAM=$DB             ;ADDRESS OF FILE NAME HIGH BYTE
0027  0000              LD=$D2                ;LOGICAL DEVICE NUMBER
0028  0000              PA=$D4                ;PRIMARY ADDRESS
0029  0000              SA=$D3                ;SECONDARY ADDRESS
0030  0000              FILEN=$D1             ;LENGTH OF FILE NAME
0031  0000              ST=$96                ;STATUS CODE
0032  0000              BOTT=$83C0            ;LEFT BOTTOM SCREEN
0033  0000                                   ;
0034  0000                                   ;
0035  0000                                   ; SYSTEM SUBROUTINE CALLS
0036  0000                                   ;
0037  0000                                   ;
0038  0000              TESTOP=$F301          ;TEST STOP
0039  0000              GETKEY=$FFE4          ;GET KEY PRESS
0040  0000              READY=$C38B           ;READY !!!!!!
0041  0000              INPUT=$C46F           ;INPUT FROM KEYBOARD
0042  0000              FOPEN=$F524           ;OPEN FILE LA,FA,SA
0043  0000              CHKIN=$F770           ;SET INPUT DEVICE
0044  0000              INBYTE=$F18C          ;INPUT SOURCE BYTE
0045  0000              FCLOSE=$F2AE          ;CLOSE FILE
0046  0000              CLRCHN=$FFCC          ;CLOSE I/O CHANNELS
0047  0000              SCROLL=$E53F          ;SCROLL SCREEN
0048  0000                                   ;
0049  0000                                   ;

SEQSRC......PAGE 0002

LINE# LOC   CODE         LINE


0051  0000                                   ;
0052  0000                    *=$7DB1;BEGIN HERE
0053  7DB1                                   ;
0054  7DB1                                   ;
0055  7DB1                                   ;** NOTE THAT TOP OF MEMORY MUST
0056  7DB1                                   ;** FIRST BE LOWERED TO PROTECT
0057  7DB1                                   ;** THIS ROUTINE AND ALLOW IT TO
0058  7DB1                                   ;** BE USED AS A UTILITY WHILST
0059  7DB1                                   ;** PROGRAMMING.
```

Continued on Page 21.

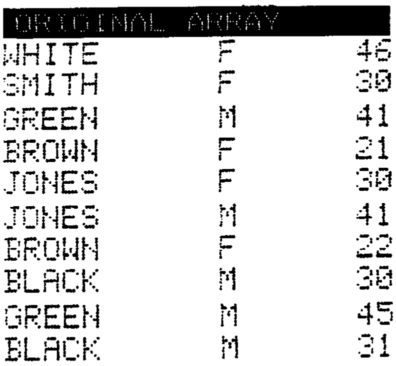
Multi Key Sorting

Mike Gross-Niklaus

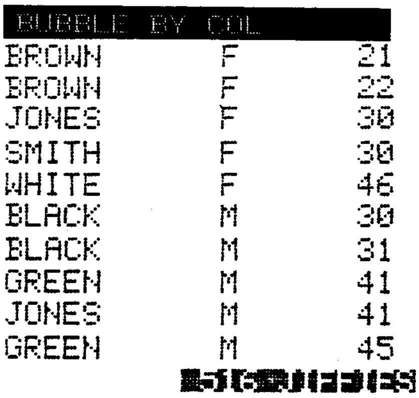


Mike Gross-Niklaus, this month sorting out sorts

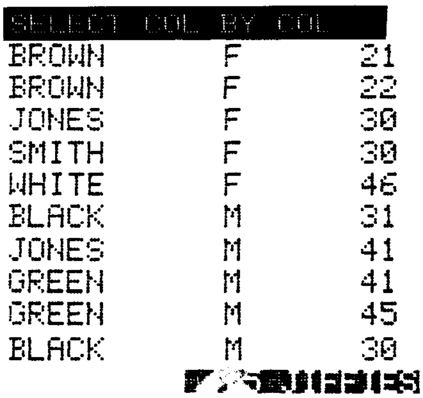
1. Sorting column by column. When sorting a two dimensional array using two or more columns as primary and secondary keys, many of us would chose to do it column by column. For example, if you have a table containing NAME in column 1, SEX male or female in column 2 and AGE in column 3, then to sort the array into "NAME within AGE within SEX", most people would do a sort on column 1, then on column 3 and finally on column 2.



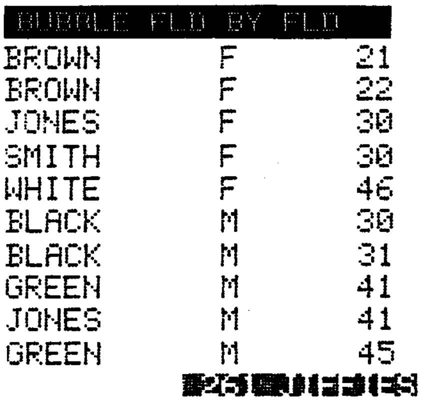
```
ORIGINAL ARRAY
WHITE        F      46
SMITH        F      30
GREEN        M      41
BROWN        F      21
JONES        F      30
JONES        M      41
BROWN        F      22
BLACK        M      30
GREEN        M      45
BLACK        M      31
```



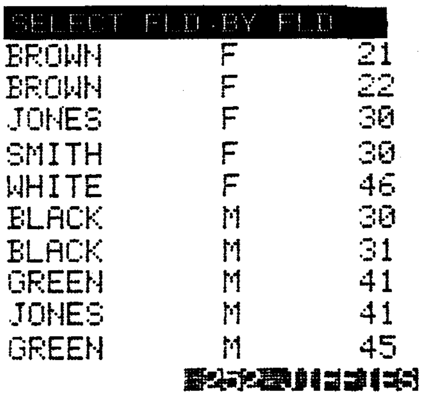
```
BUBBLE BY COL
BROWN        F      21
BROWN        F      22
JONES        F      30
SMITH        F      30
WHITE        F      46
BLACK        M      30
BLACK        M      31
GREEN        M      41
JONES        M      41
GREEN        M      45
```



```
SELECT COL BY COL
BROWN        F      21
BROWN        F      22
JONES        F      30
SMITH        F      30
WHITE        F      46
BLACK        M      31
JONES        M      41
GREEN        M      41
GREEN        M      45
BLACK        M      30
```



```
BUBBLE FLD BY FLD
BROWN        F      21
BROWN        F      22
JONES        F      30
SMITH        F      30
WHITE        F      46
BLACK        M      30
BLACK        M      31
GREEN        M      41
JONES        M      41
GREEN        M      45
```



```
SELECT FLD BY FLD
BROWN        F      21
BROWN        F      22
JONES        F      30
SMITH        F      30
WHITE        F      46
BLACK        M      30
BLACK        M      31
GREEN        M      41
JONES        M      41
GREEN        M      45
```

```
READY.
```

Some sorts, however, such as the 'Select', sorting column by column will not give you the desired order. And that's a pity because the 'Select' sort, (and the Shell-Metzner of course), can be faster than a Bubble sort.

2. Sorting field by field.

An alternative method to column by column sorting is to do the sort just once, but comparing as many fields per element as are necessary to determine the correct order. Taking the example above, the sort would for each pair of elements, look at the SEX field, then if both fields were the same, at the AGE, and if they were equal then the NAME. Using this scheme, multi-key sorting works using any of the common sorting routines.

3. A demonstration

Shown below is a demonstration program which sorts the same array of data, using BUBBLE and SELECT sorts, in both 'column by column' and 'field by field' modes. I've laid out the program in my usual style, but spaced out the statements on to a line with plenty of REM statements to help you see what's going on.

```
10 REM MULTI-KEY SORT DEMO
20 REM MIKE GROSS-NIKLAUS
30 REM 9.5.81
40 REM FOR CPUCN
99 REM"

1000 REM PRELIMINARIES
1010 DIM DA$(10,3), CO(3)
1099 REM"

2000 REM DO FOUR SORTS
2010 GOSUB 9000          :REM SETUP
2020 PRINT"▮▮ORIGINAL ARRAY"
2022 GOSUB 8000          :REM ARRAY PRINT
2030 CO(1)=1 :CO(2)=3 :REM COL BY
2032 CO(3)=2             :REM  COL
2040 GOSUB 3000          :REM BUBBLE
2050 GOSUB 4000          :REM INSERT
2060 CO(1)=2 :CO(2)=3 :REM FLD BY
2062 CO(3)=1             :REM  FLD
2070 GOSUB 5000          :REM BUBBLE
2080 GOSUB 6000          :REM SELECT
2090 END
2099 REM"

READY.
```

In the Preliminaries block the Data array which is to be sorted is set up as DA$(10,3). The small CO(3) array is used to hold the pointers to the next sort column in the correct order of processing.

In the main block, starting at 2000, the original data is fed into the array and displayed in unsorted sequence. Then the column priorities are set up and each sort obeyed. Notice that the column priorities are different for the two modes. Column by column sorts require the columns specifying in reverse order of importance. Field by field sorting requires them in priority order.

```
3000 REM BUBBLE COL BY COL
3010 PRINT"▮BUBBLE BY COL"
3012 T1 = TI                :REM STRTTIME
3020 FOR PR = 1 TO 3        :REM COLS
3022 FOR LM = 9 TO 2 STEP -1
3024 SW= 0                  :REM SWOPFLG
3026 FOR RO = 1 TO LM       :REM EACH
3028 A$= DA$(RO,CO(PR))     :REM  ELMNT
3030 B$= DA$(RO+1,CO(PR)):REM   PAIR
3032 IF A$<=B$ THEN 3036  :REM SKIP
3034 X= RO+1   GOSUB 7000:REM SWOP
3036 NEXT RO                :REM NXT PR
3038 IF SW=0 THEN 3042      :REM COL END
3040 NEXT LM                :REM NXT PASS
3042 NEXT PR                :REM NXT COL
3050 T2 = TI                :REM STOPTIME
3052 GOSUB 8000             :REM DISPLAY
3054 RETURN
3099 REM"

4000 REM SELECT COL BY COL
4010 GOSUB 9000 : PRINT"▮SELECTCOL BY COL"
4012 T1 =TI                 :REM STRTTIME
4020 FOR PR = 1 TO 3        :REM COLS
4022 FOR LM = 1 TO 9
4024 FOR RO = LM+1 TO 10  :REM MATCH
4026 A$ =DA$(LM,CO(PR))    :REM  TWO
4028 B$= DA$(RO,CO(PR))    :REM   ELMNTS
4030 IF A$<=B$ THEN 4034  :REM SKIP
4032 X=LM : GOSUB 7000     :REM SWOP
4034 NEXT RO                :REM NXT PR
4036 IF SW=0 THEN 4040      :REM COL END
4038 NEXT LM                :REM SHORTEN
4040 NEXT PR                :REM NEXT COL
4050 T2 = TI                :REM STOPTIME
4052 GOSUB 8000             :REM DISLPLAY
4054 RETURN
4099 REM"

READY.
```

The first sort, in block 3000, is the ubiquitous Bubble, nested into a loop, PR, to define which column should be sorted next. The code for swopping over all three fields of the two elements being processed has been placed in a subroutine, 7000,

```
5000 REM BUBBLE FIELD BY FIELD
5010 GOSUB 9000 : PRINT"▮BUBBLE FLD BY FLD"
5012 T1= TI                 :REM STRTTIME
5020 FOR LM = 9 TO 2 STEP -1
5022 SW= 0                  :REM SWOPFLAG
5024 FOR RO = 1 TO LM       :REM EACH
5026 FOR PR = 1 TO 3        :REM  FIELD
5028 A$= DA$(RO,CO(PR))     :REM   ELMNT
5030 B$= DA$(RO+1,CO(PR)):REM    PAIR
5032 IF A$<B$ THEN 5042   :REM SKIP
5034 IF A$=B$ THEN 5040   :REM FLDS =
5036 X= RO+1 : GOSUB 7000:REM SWOP
5038 GOTO 5042              :REM & NXT PR
5040 NEXT PR                :REM NXT FLD
5042 NEXT RO                :REM NXT PAIR
5044 IF SW = 0 THEN 5050  :REM FINISH
5046 NEXT LM                :REM NXT PASS
5050 T2= TI                 :REM STOPTIME
5052 GOSUB 8000             :REM DISPLAY
5054 RETURN
5099 REM"

6000 REM SELECT FIELD BY FIELD
6010 GOSUB 9000 : PRINT"▮SELECT FLD BY FLD"
6012 T1= TI                 :REM STRTTIME
6020 FOR LM = 1 TO 9
6022 FOR RO = LM+1 TO 10:REM MATCH
6024 FOR PR = 1 TO 3       :REM  FLD
6026 A$= DA$(LM,CO(PR))    :REM   ELMNT
6028 B$= DA$(RO,CO(PR))    :REM    PAIR
6030 IF A$<B$ THEN 6040   :REM SKIP
6032 IF A$=B$ THEN 6038   :REM FLDS =
6034 X= LM : GOSUB7000     :REM SWOP
6036 GOTO6040               :REM & NXT PR
6038 NEXT PR                :REM NXT FLD
6040 NEXT RO                :REM NXT PAIR
6042 NEXT LM                :REM NXT PASS
6050 T2= TI                 :REM STOPTIME
6052 GOSUB 8000             :REM DISPLAY
6054 RETURN
6099 REM"

READY.
```

because all four sorts require elements to be swopped. The start finish times are placed in T1 and T2, and a call to the display routine in 8000 allows you to see that the Bubble has done its stuff, and tells you how many jiffies it took.

The second sort in 4000, is a 'Select' sort using column by column mode. First the array is restored to it's original unsorted order, by calling the setup routine in 9000. The sort works by comparing every element in turn with element 1, and swopping if appropriate. The element 2 is treated the same way and so on. Unfortunately, while it works fine on one-key sorts, in this case, when the sorted array is printed you will see that it has failed to achieve the desired order!

The sort in block 5000 is the Bubble again, this time using field by field mode. After restoring the original array, the column loop, PR, is this time nested within the RO loop which specifies the elements being compared. PR selects the fields in the correct order, only moving on to the next field when the previous one has resulted in an exact match.

The final sort in block 6000 is the 'Select' used in field by field mode.

Again the PR loop is nested within the Row selection loop, RO.

```
7000 REM SWOP ELEMENTS
7010 FOR SC = 1 TO 3
7020 F$ = DA$(RO,SC)
7030 DA$(RO,SC) = DA$(X,SC)
7040 DA$(X,SC) = F$
7050 NEXT SC : SW = 1 : RETURN
7099 REM"
```

```
8000 REM PRINT ARRAY
8020 FOR RO = 1 TO 10
8030 FOR CO = 1 TO 3
8040 PRINT DA$(RO,CO),
8050 NEXT CO : PRINT
8060 NEXT RO
8070 IF EF=0 THEN 8090
8080 PRINT "              ▮"T2-T1; "JIFFIES"
8090 EF=1 : RETURN
8099 REM"

9000 REM SET UP ARRAY
9010 RESTORE
9020 FOR RO = 1 TO 10
9030 FOR CO = 1 TO 3
9040 READ DA$(RO,CO)
9050 NEXT CO, RO
9060 RETURN
9099 REM"

10000 REM SAMPLE DATA FOR THE SORT
10010 DATA WHITE,F,46
10020 DATA SMITH,F,30
10030 DATA GREEN,M,41
10040 DATA BROWN,F,21
10050 DATA JONES,F,30
10060 DATA JONES,M,41
10070 DATA BROWN,F,22
10080 DATA BLACK,M,30
10090 DATA GREEN,M,45
10100 DATA BLACK,M,31

READY.
```

Blocks 7000, 8000 and 9000 contain subroutines for Swopping, Printing the array, and filling up the array from Data statements respectively. Block 10000 contains some sample data.

Some sample results using the test data are shown in the printouts. Obviously the 'Select by column' has failed, while 'Select using fields' has worked. The times of the two Bubble sorts indicate that field mode is considerably quicker than sorting by column, and this difference should become more and more pronounced as the size of the sort increases.

4. Applications.

I've illustrated this article with the Bubble and Selection sorts because they are the easiest to follow, and it's the principle, rather than clever sorting, that is the point here. You may like to try applying the scheme to the Shell-Metzner, (versions of which have been published in this magazine), which is faster than either the Bubble or the Selection sort.

It's worth noting that many of the machine sorts now commercially available use the 'field' method described here to accomplish multi-key sorting. Those of you who are into assembler code will find that the principle can be adapted to the sort shown in an earlier volume of CPUCN, published a couple of months ago.

Mike Gross-Niklaus

Disk Use for Beginners

David J. Pocock

Hello again! Unfortunately I must start this months article with an apology. Last month I said that there was no difference between the 2040 and 3040, but this is not quite true. The operation of the two units is the same, certainly as far as programming them is concerned; however there is a difference in the 'Main Board' inside the unit. This difference only becomes apparent if you want to upgrade from DOS 1.2 (2040/3040) to DOS 2.1 (4040). Converting a 3040 to a 4040 is simply a matter of changing ROMs on the main board, but to change a 2040 to a 4040 involves changing the 'Main Board'.

With that out of the way, here goes for PART 2. This month :-
DUPLICATE, LOAD, SAVE, VERIFY, COPY, the COMMAND CHANNEL and ERRORS.

WARNING : As we are doing much more serious stuff this month do not attempt any of the commands until you have finished reading the article.

DUPLICATE

Before you go any further insert the Utility Disk into drive 0 (left) and a blank disk into drive 1 then CAREFULLY type the following:-

```
OPEN 1,8,15
PRINT 1,"D 1 = O" (no spaces)
```

Ensure that you have inserted the disks in the correct drives. The disks should both spin for 3-4 mins (8 for 8050s) and the LEDs on the drives will both be lit (the central LED should not be on (green for 8050)). At the end of this period remove the Utility Disk from drive 0 and put it away safely. Now remove the disk from drive 1 and re-insert the disk in drive 0.

What you have just done is to make a copy of the Utility Disk onto a blank disk.

If the centre LED glows RED make sure that the disks are in the correct drives, and then try it once more. If it fails again see your dealer and tell him that your Utility disk is corrupt.

NOTE :- If you get the disks in the wrong drives you may end up copying a blank disk onto the Utility Disk.

LOAD

Before we can SAVE a program we must first of all have a program to SAVE. For those of you who know any BASIC you can write your own small program (anything, even just printing 'HELLO'). Otherwise you can LOAD an existing program off the Utility Disk. Pick any of the programs on the utility disk (those with PRG against the name not SEQ), then load the program :-

```
LOAD"x:name",8
```

where

x is the drive number (0 or 1).
name is the Program name.
8 is the disk device number.

LIST to show something has loaded. Now that you have done that take another blank disk and place it in drive 1. Format the disk (see last months article on formatting new disks).

SAVE

Now that you have a program in the PET you will want to store it on the disk for later use. To do this type :-

```
SAVE "x:name",8
```

Save the program onto the blank (formatted) disk 1, in which case x becomes 1.

If you miss the drive number out the program will not be saved correctly, and garbage will be written onto the disk. If this happens you must delete the program from the disk (check both) and tidy up the disks (See next month).

VERIFY

As with cassettes you will want to check that the information stored on the disk is correct. To do this type :-

```
VERIFY"x:name",8
```

x and name are the same for SAVE. If the PET responds with 'VERIFY ERROR' delete the file and re-save. If the information is correct the PET will respond with OK.

COPY and the **COMMAND CHANNEL**

This command enables you to copy a file (PRG or SEQ. USR files can be copied on 4040s and 8050s but not 3040s) from one drive to the other without it passing through the PET. For those of you with more than one disk unit (there are some!) it cannot be used to copy from one disk unit to another.

The syntax (i.e. the structure for the COPY command is :-

```
C x:new = y:old (no spaces)
```

where

C Stands for COPY.
x is the destination drive number.
new is the destination file name.
y is the source drive number.
old is the source file name.

To use the COPY command you must open the command channel. You have done this before when you initialised the disk, formatted the disk and when you duplicated a disk. To open the command channel :-

```
OPEN f,8,c
```

where

f is the file number (1-255).
8 is the disk divice number.
c is the channel (15 = Command Channel).

Normally I use f = c as this reminds me what the file is for and saves confusion when commands use 'f' and 'c' for different things (a later article).

The secondary address or channel 'c' has various values for different uses. Its value is 15 for the command channel.

To send a copy command :-

```
PRINT 1, "Cx:new=y:old"
```

You can send as many commands as you wish until you close the command channel with 'CLOSE 1' or you LOAD, SAVE or VERIFY a program.

ERRORS

The command channel has another use apart from telling the disk unit what you want it to do. The command channel is also used by the disk unit to leave messages that will tell you that things have worked correctly or what has gone wrong.

To access these 'ERROR' messages you must read from the command/error channel. This is done using the INPUT Command. As INPUT can only be used in a program, below is a program which reads and prints the 'ERROR' message.

```
NEW
10      OPEN 15,8,15
40      INPUT 15,EN,EM$,ET,ES
50      PRINT EN; EM$; ET; ES
70      CLOSE 15
```

The message consists of four parts :-

EN The Error Number. This number gives an accurate error report. Details of what each number means is in the disk manual.

EM$ The Error Message. This string gives a brief description of the error, for example 'READ ERROR'. This tells you that the disk unit could not read part of a disk for some reason. A more detailed reason is found in the disk manual under the correct error number. (Note that there are 6 different READ ERRORs).

ET The Error Track. This tells you the track on which the error occured.

ES The Error Sector. This tells you where on the track the error occured.

Some of the messages are not associated with a particular TRACK and SECTOR and so these parts of the message are set to zero

When you delete files from the disk the message left tells you how many files have been erased in the Error Track part of the message (more on that next time).

If you add the following lines to the program above it will give a short utility program to use while experimenting.

```
20 INPUT "COMMAND * (CLR)(CLR)(CLR)";CM$:IF CM$="*" THEN 70
30PRINTS 15, CM$
60 GOTO 20
```

This program will now ask for a command string, send the command to the disk and then print the 'ERROR' message. To finish just press RETURN when asked for the command.

e g s .

COMMAND ? C1:PROGRAM2=0:PROGRAM1

This will copy PROGRAM from drive 0 and put it on drive 1 and call it PROGRAM2.

While we're still here I suggest that you SAVE this program for later.

THATS ALL (for this time) FOLKS.

Next Time :- SCRATCH, VERIFY, RENAME, SHORT HAND (and MORE ???)

Continued from page 17

```
0060  7DB1                                     ;**
0061  7DB1                                     ;** DO THIS WITH:-
0062  7DB1                                     ;**                    POKE 52,176
0063  7DB1                                     ;**                    POKE 53,125
0064  7DB1                                     ;**
0065  7DB1                                     ;** THE ROUTINE WILL THEN
0066  7DB1                                     ;** CO-EXIST WITH DOS-SUPPORT.
0067  7DB1                                     ;
0068  7DB1                                     ;
0069  7DB1   A9 0E              LDA #$0E       ;LOAD ACC WITH 14
0070  7DB3   8D 4C E8           STA $E84C      ;STORE IN 59468
0071  7DB6   A9 3E              LDA #$3E       ;LOAD ACC WITH 62
0072  7DB8   8D 62 E8           STA $E862      ;STORE IN 59490
0073  7DBB                                     ;
0074  7DBB   20 6F C4   START   JSR INPUT       ;GET FILE NAME
0075  7DBE   A9 00              LDA #$00       ;LOW BYTE OF FILENAME ADDRESS
0076  7DC0   85 DA              STA LONAM      ;STORE IN POINTER LOW
0077  7DC2   A9 02              LDA #$02       ;HIGH BYTE OF FILENAME ADDRESS
0078  7DC4   85 DB              STA HINAM      ;STORE IN POINTER HIGH
0079  7DC6   A9 02              LDA #$02       ;SET LOGICAL FILE NO:2
0080  7DC8   85 D2              STA LD         ;STORE IN LOG. DEV. NO:POINTER
0081  7DCA   A9 08              LDA #$08       ;SET DEVICE NO:8
0082  7DCC   85 D4              STA PA         ;STORE IN DEV. NO:POINTER
0083  7DCE   A9 02              LDA #$02       ;SET SECONDARY ADDR.2
0084  7DD0   85 D3              STA SA         ;STORE IN SEC. ADDR. POINTER
0085  7DD2   A2 00              LDX #$00       ;SET INDEX TO ZERO
0086  7DD4                                     ;
0087  7DD4   BD 00 02   LOOP1   LDA $0200,X     ;GET CHARACTER FROM INPUT BUFFER
0088  7DD7   F0 07              BEQ EXEC       ;IF ZERO THEN FILENAME END
0089  7DD9   E8                 INX            ;BUMP UP INDEX
0090  7DDA   20 01 F3           JSR TESTOP     ;TEST STOP KEY
0091  7DDD   4C D4 7D           JMP LOOP1      ;AND DO AGAIN
0092  7DE0                                     ;
0093  7DE0   86 D1      EXEC    STX FILEN       ;SET FILENAME LENGTH
0094  7DE2   20 24 F5           JSR FOPEN      ;OPEN FILE
0095  7DE5   A2 02              LDX #$02       ;SET X REGISTER TO LOG FILE NUMBER
0096  7DE7   20 70 F7           JSR CHKIN      ;SET INPUT DEVICE
0097  7DEA   A2 00              LDX #$00       ;SET X REGISTER TO ZERO
0098  7DEC   A0 00              LDY #$00       ;SET Y REGISTER TO ZERO
0099  7DEE                                     ;
0100  7DEE   8E 00 10   LOOP2   STX $1000       ;SAVE X VALUE
0101  7DF1   20 01 F3           JSR TESTOP     ;TEST FOR STOP KEY
0102  7DF4   F0 03              BEQ WAIT       ;YES ?...THEN HOLD IT!
0103  7DF6   4C 03 7E           JMP CONT       ;NO ?...CONTINUE
0104  7DF9                                     ;


SEQSRC......PAGE 0003

LINE# LOC   CODE          LINE


0106  7DF9   20 E4 FF   WAIT    JSR GETKEY      ;LOOK FOR ANY KEY
0107  7DFC   F0 FB              BEQ WAIT       ;NO ?...THEN STILL WAIT
0108  7DFE   A2 02              LDX #$02       ;RESET X REGISTER
0109  7E00   20 70 F7           JSR CHKIN      ;..AND RE-CHECKIN
0110  7E03                                     ;
0111  7E03   A9 00      CONT    LDA #$00        ;CLEAR ACC ??
0112  7E05   20 8C F1           JSR INBYTE     ;GET CHARACTER FROM FILE
0113  7E08   A4 96              LDY $96        ;TEST STATUS
0114  7E0A   D0 08              BNE END        ;END OF FILE OR ERROR DETECTED
0115  7E0C   AE 00 10           LDX $1000      ;RECOVER X REGISTER
0116  7E0F   A0 00              LDY #$00       ;CLEAR Y REGISTER ??
0117  7E11   4C 23 7E           JMP LOOP3      ;PRINT CHARACTER
0118  7E14                                     ;
0119  7E14   A2 00      END     LDX #$00        ;CLEAR X REGISTER ??
0120  7E16   A0 00              LDY #$00       ;CLEAR Y REGISTER ??
0121  7E18   A5 D2              LDA LD         ;LOAD LOGICAL FILE NUMBER
0122  7E1A   20 AE F2           JSR FCLOSE     ;CLOSE FILE
0123  7E1D   20 CC FF           JSR CLRCHN     ;CLEAR CHANNELS &
0124  7E20                                     ;RESET DEFAULT I/O
0125  7E20   20 8B C3           JSR READY      ;BACK INTO COMMAND BASIC
0126  7E23                                     ;
0127  7E23   A8         LOOP3   TAY             ;TRANSFER ACCUM. TO Y-INDEX
0128  7E24   C0 0D              CPY #$0D       ;COMPARE TO CARRIAGE RETURN
0129  7E26   D0 08              BNE PRINT      ;NO ?...THEN PRINT IT!
0130  7E28                                     ;
0131  7E28   20 3F E5   SCRL    JSR SCROLL      ;ELSE SCROLL SCREEN
0132  7E2B   A2 00              LDX #00        ;RESET X-REGISTER
0133  7E2D   4C EE 7D           JMP LOOP2      ;AND RETURN ...
0134  7E30                                     ;
0135  7E30   98         PRINT   TYA             ;TRANSFER Y-REGIS TO ACCUM.
0136  7E31   9D C0 83           STA BOTT,X     ;PRINT CHARACTER TO SCREEN
0137  7E34   E8                 INX            ;INCREMENT X-REGISTER
0138  7E35   E0 28              CPX #$28       ;COMPARE TO 40
0139  7E37   90 B5              BCC LOOP2      ;NO ?...THEN RETURN
0140  7E39   4C 28 7E           JMP SCRL       ;ELSE SCROLL SCREEN
0141  7E3C                                     ;
0142  7E3C                      .END


ERRORS = 0000


SYMBOL TABLE

SYMBOL  VALUE
 BOTT     83C0    CHKIN    F770    CLRCHN   FFCC    CONT     7E03
 END      7E14    EXEC     7DE0    FCLOSE   F2AE    FILEN    00D1
 FOPEN    F524    GETKEY   FFE4    HINAM    00DB    INBYTE   F18C
 INPUT    C46F    LD       00D2    LONAM    00DA    LOOP1    7DD4
 LOOP2    7DEE    LOOP3    7E23    PA       00D4    PRINT    7E30
 READY    C38B    SA       00D3    SCRL     7E28    SCROLL   E53F
 ST       0096    START    7DBB    TESTOP   F301    WAIT     7DF9


END OF ASSEMBLY
```

PET and the IEEE Bus

1 The operation of the IEEE or HP Interface Bus is generally admitted to be somewhat complicated. However it is now a well established industry standard for scientific and techinical application and is one of the unique features of the Commodore PET system.

As well as the 'IEEE-488 Standard Digital Interface for Programmable Instrumentation - 1978', which was a revision of the 1975 Standard, a great deal has been written about the working of this bus, both by the Hewlett Packard people who originated the protocol, and by enthusiasts for the PET interested in interfacing it to the many IEEE compatible instruments, such as programmable power supplies and digital voltmeters, which are now available. In addition an excellent example of the bus protocol implemented in machine code has already been published in CPUCN. No. 04.

This article attempts to examine the actual machine code ROM routines employed in the original 2001 version of the PET. A number of articles, and a book published recently, have given detailed information on the IEEE bus, interpreting the operation of BASIC commands and statements with reference to the bus, but little appears to have been shown of the machine code or assembler mnemonic routines which actually implement the protocol. The register bit manipulations which activate the management and transfer bus signals are detailed here together with the corresponding register addresses.

IEE ROM routines

The IEEE ROM routines can be obtained by disassembling the code from $F0B6 on to $FICB in the upper ROM area of the operating system, and are reproduced, with comments, in the Appendix Pages 1-3. These are easily found as they follow immediately after a number of File Messages ending with 'READY', and are followed by the routines which execute the BASIC commands for CMD devices. A (hopefully) complete list of IEEE-488 Register Addresses is given in Table 1 along with the addresses of those locations which are repeatedly used by the IEEE routines in Table 2. Return entry points are given in Table 3.

Since many detailed descriptions of the IEEE bus signals are given elsewhere only a summary will be given below. It should be noted that negative logic is used on the IEEE bus so that bus signals and commands are active low-true, and high-false, and that data bits are inverted on the bus i.e. it is necessary to complement data bits transmitted or received on the bus and to leave the bus in an active high state.

⟶

Pet finding out the latest test score

Apologies section

In last month's article by Mike Gross-Niklaus, the section headed A Counting Algorithm contained a mistake. Sorry...Step B should have read "Pick up the RIGHT hand digit and add one to it", not the left hand digit. The same error also occured in describing the ADDONE routine. Mike's explanation was that after spending so many years giving talks and demonstrations in front of people he is used to thinking of right as left and vice versa for the benefit of people looking at him. Seems reasonable to me.

Apologies to John Collins who occasionally goes boldly rather than bodly.

No doubt there are other spelling mistakes but I don't think there was anything disastrous.

Finally, a mention for the North London Computer Club (rah rah!!) who supplied the pictures of John Collins and Barry Miles in last month's issue but who didn't get an acknowledgment themselves. Thank you North London Computer Club - you're doing a grand job, keep it up.

The same message applies to all other user (usr?) groups everywhere - when the Supreme Being gets round to producing a day that's got a lot more than 24 hours in it I'll be getting in touch with you.

Original Implementation

The original 2001 PET implementation of the IEEE-488 Bus is a subset of the standard suitable got most purposes (this has been rectified in the new ROM sets). In a PET system, the PET can be the only controller on the bus. Of the Interface Management Lines - REN is held permanently grounded by the PET (enabled), IFC is driven low-true during the reset sequence, and SRQ is not implemented from BASIC but accessible from machine code. ATN is asserted low-true during addressing and command sequences otherwise is high-false for data. EOI is always pulled low-true by the controller during the transfer of its last data byte on the DIO lines. Of the Transfer Control Lines (handshake lines) the talker originates DAV and listeners originate NRFD and NDAC. The handshake signals are:-

NRFD (Not Ready For Data) - active low signal line indicates that one or more assigned listeners are not ready to receive the next data byte - when all the assigned listeners for a particular data transfer have released NRFD, the NRFD line goes inactive - high - this tells the talker to place the next data byte on the Data Bus.

DAV (Data Valid) - line is activated by talker shortly after talker places a valid data byte on the data bus - an active low DAV signal tells each listener to capture the data byte presently on the bus - the talker is inhibited from activating DAV when NRFD is active low.

NDAC (Not Data Accepted) -signal line is held active low by each listener until the listener captures the data byte then NDAC goes inactive high - this tells the talker to take the byte off the Data Bus.

The PET uses ASCII codes to transmit and receive characters on the bus, and an extended version of these ASCII codes to transmit primary and secondary addresses and other control information on the IEEE bus.

Any instruments on the bus must have a device number in the range 4 to 30 (0-3 are used by the KYBD, cassettes 1 and 2 and video screen respectively) which is usually set by DIL switches in the instrument, and the instrument must be addressed accordingly. This is done by transmitting on the bus the primary address evaluated from the Talk or Listen Address Group. The formats are TAG = 0100 AAAA and LAG = 0010 AAAA with MTA = \$40 (sets bit 6) and MLA = \$20 (sets bit 5) and evaluated as:-

TAG = (MTA) OR (Device No);
LAG = (MLA) OR (Device No).

The secondary address formats are SCG = 0110 SSSS, 1110 SSSS, 1111 SSSS with the latter forms used with 'OPEN' or 'CLOSE'.

The universal command Unlisten and Untalk are:-

UNL = \$3F = 0011 1111, UNT = \$5F = 0101 1111,

the lower nibble having the effect of overwriting any data bits on the bus. As to the assembler codes used, the 6502 mnemonic BIT instruction is worth noting as this sets both the N and V flags in the P register as well as the Z flag, i.e.:-

BIT : AM, M_7 — N, M_6 — V and if AM = 0 then Z = 1

which is a most powerful and useful instruction.

Typical Transaction

In a typical bus transaction, proceeded by an 'OPEN' statement in BASIC, the PET controller must take command of the bus and output the appropriate Listen (LAG) or talk (TAG) command to assign devices as listeners or talkers on the bus. Meanwhile PET indicates that command or control characters are valid on the bus by putting ATN low-true. However PET must first ensure that bytes remaining (deferred) in the IEEE output buffer from a previous transaction are transmitted on the bus, along with EOI, before this is done. In general PET transmits the previous character (deferred char.) and stores the current character in the IEEE output buffer, thus delaying transmission at all times by one byte. NRFD and NDAC are normally held high-false so that unless NDAC is pulled low by an active device on the bus a "Device Not Present" error will result - indicated by setting bit 7 of the status byte (ST =-128). A flowchart of the "Send TALK/LISTEN to IEEE" routine is given in Fig 1 and the Rom routine in appendix page 1.

Send TALK/LISTEN to IEEE - routine stores MTA or MLA on the stack, generates the appropriate bus signals, then tests the deferred output flag byte and as a result either transmits the last byte with EOI set low-true, followed by the TAG or LAG (evaluated from the MTA/MLA values retrieved from the stack), or else sends TAG/LAG directly.

To actually transmit the byte and effect the handshake protocol PET enters the "Send Byte to IEEE" routine shown in Fig 2 and appendix page 1.

Send BYTE to IEEE - in this routine PET is controller/talker and so originates DAV and expects to receive RFD and DAC - NDAC must go high within the 65 ms time out limit or the timer times out and the status bit is set for time out on WRITE - this routine gets the deferred data character from the IEEE output buffer then handles the handshake signal (as shown in the timing diagram Fig 2a) and so transmits the character on the bus.

To get a byte or character from the IEEE bus PET enters the routine 'GET Byte from IEEE' shown in Fig 3 and page 3 which is essentially similar to the output routine.

Get BYTE from IEEE - in this routine PET is controller/listener and so originates NDAC and NRFD and expects to receive DAV - again the timer is reset and tested for the time out on READ so that DAV must go low-true within 65 ms of NRFD being set high-false - PET performs the input handshake with the timing as shown in Fig 3a and returns the data byte in the accumulator.

The ROM routines which generate the Secondary Address for Talker/Listener and the Untalk/Unlisten routines are shown on page 2 of the appendix.

It is hoped that the above will provide some insight into how the PET ROM routines implement the IEEE-488 bus protocol. However I am much indebted to the many people, including Fisher and Jensen, Ron Geere and Mike Todd, Nick Hampshire, Dr C Preece, John Cooke and Gregory Yob, who have previously published material on the IEEE bus or indicated the address locations in ROM. I would also be grateful to anyone pointing out any errors, inaccuracies and omissions (of which there must by many), as this will contribute to the general understanding of how the bus is operated in the PET.

DAVID MUIR
Department of Physics
Napier College
Merchiston
EDINBURGH

Continued over

TABLE 1
IEEE-488 Hardware Addresses and Signal Information

Hardware:- PIA 652Ø #1 (KYBD PIA), PIA 652Ø #2 (IEEE PIA), VIA 6522

HARDWARE	SIGNAL SYMBOLIC	HEX	DEC	BITS	IEEE	SIGNAL	I/O
LOCATION	NAME	ADDRESS	ADDRESS	USED	SIGNAL	MODE	LINES
652Ø#2	IEEIØ-7	E82Ø	59424	Ø-7	DIO1-8	INPUT	PAØ-7
652Ø#2	IEEOØ-7	E822	59426	Ø-7	DIO1-8	OUTPUT	PBØ-7
652Ø#2	IEEIS	E821	59425	3	$\overline{\text{NDAC}}$	OUTPUT	CA2
				7	$\overline{\text{ATN}}$	INPUT	CA1
652Ø#2	IEEOS	E823	59427	3	$\overline{\text{DAV}}$	OUTPUT	CB2
				(bits 4,5 = 1)			
				7	$\overline{\text{SRQ}}$	INPUT (not buffered)	CB1
652Ø#1	PIAL	E81Ø	594Ø8	6	$\overline{\text{EOI}}$	INPUT	PA6
652Ø#1	PIAL1	E811	594Ø9	3	$\overline{\text{EOI}}$	OUTPUT	CA2
				(bits 4,5 = 1)			
6522	PIAØ	E84Ø	59456	Ø	$\overline{\text{NDAC}}$	INPUT	PBØ
	PIA1			1	$\overline{\text{NRFD}}$	OUTPUT	PB1
	PIA2			2	$\overline{\text{ATN}}$	OUTPUT	PB2
	PIA6			6	$\overline{\text{NRFD}}$	INPUT	PB6
	PIA7			7	$\overline{\text{DAV}}$	INPUT	PB7
6522	T1C-H	E845	59461	Timer for timeout on READ/WRITE (enter #FF to reset)			
6522	IFR	E84D	59469	6	Register to control timeout		IFR

TABLE 2
IEEE-488 Adresses

Address	Location/Contents/Function
ØØEF	Current logical file number.
ØØFØ	Current secondary address
	(has bit 5 and bit 6 set-defaults to FF)
ØØF1	Current device number.
ØØFD (253)	Timeout status bit.
Ø2ØC (524)	Status Byte (BASIC variable ST)
	bit Ø AND mask $Ø1 Timeout on data transfer from PET (WRITE)
	i.e. no response on NDAC line for 65 ms from listener
	if NDAC is low-true bit Ø is set (ST = 1).
	bit 1 AND mask $Ø2 Timeout for data transfer into PET (READ)
	i.e. no response on $\overline{\text{DAV}}$ line for 65 ms from talker
	if $\overline{\text{DAV}}$ is high-false bit 1 is set (ST = 2)
	bit 6 AND Mask $4Ø EOI signal (ST = 64)
	bit 7 AND mask $8Ø Device not present (ST = 128)
Ø21D (541)	IEEE deferred output flag byte
	(contains FF if character waiting to the output, ØØ if none)
Ø222 (546)	IEEE output buffer
	(used to delay IEEE output by one character)
Ø263	Device no. for INPUT
	(current CMD input device - defaults to ØØ - KYBD)
Ø264	Device no. for OUTPUT
	(current CMD output device - defaults to Ø3 - SCREEN)
F13B (61755)	Flag status for TIME OUT error on WRITE
F142 (61762)	Flag status for 'Device Not Present' error
F146 (61766)	Flag status for TIME OUT error on READ
F1B5	Flag status for End or Identify (EOI) set
FBE5	Flag error in ST (set ST error flag)
	(accumulator has error bit set on entry).

TABLE 3
Routine Entry Points for IEEE-488 Bus

ROUTINE START ADDRESS	FUNCTION	ENTERED FROM
	MACHINE CODE ROUTINES	
FØB6	Send TALK (TAG) to IEEE Bus	F378, F7BC
	(Sets IEEE device (device No in ØF1) as talker)	
FØBA	Send LISTEN (LAG) to IEEE Bus	F6C6, F6EA,
	(Sets IEEE device (device no in ØF1) as listener)	F8ØD
FØBC	Set ATN true and send character in accumulator	F182
FØF1	Send BYTE (CHAR) to IEEE Bus	F172, F1F2
	(Gets a character from IEEE output buffer in Ø222	
	into accumulator and sends it on the IEEE Bus)	
F12C	Send SECONDARY ADDRESS (SCG) to listener	F6FS, F81E
	(Sends character in accumulator to IEEE listener device)	
F132	Inhibit SECONDARY ADDRESS	F164, F185,
	(Sends no secondary address if not specified)	F814
F15B	Send SECONDARY ADDRESS (SCG) to talker	F7CD
	(Sends char. in accumulator to IEEE talker device)	
F164	Inhibit SECONDARY ADDRESS	F7C3
F167	Send deferred CHAR to IEEE Bus	F244
	(Sends previous character and saves current char.)	
F17A	Send UNTALK (UNT) to IEEE Bus	F296
	(Untalk all devices on the bus)	
F17E	Send UNLISTEN (UNL) to IEEE Bus	F288
	(Unlisten all devices on the bus)	
F187	Get a BYTE (CHAR) from IEEE Bus	F22D
	(Gets a single char. from IEEE bus to accumulator)	
	BASIC ROUTINES	
F1CC	BASIC routine for 'GET' from CMD device	FFE4
F1DF	BASIC routine for 'INPUT' from CMD device	FFCF
F231	BASIC routine for 'OUTPUT' to CMD device	FFD2
F27D	BASIC routines for UNLISTEN and UNTALK all devices	
	on IEEE Bus	FFCC, CAD8
F2C8	BASIC routine for 'CLOSE'	FFE7
F52A	BASIC routine for 'OPEN'	FFCØ
F78B	Set CMD INPUT device in 263	
F7DC	Set CMD OUTPUT device in 264	

FIG 1
Send TALK/LISTEN to IEEE Bus

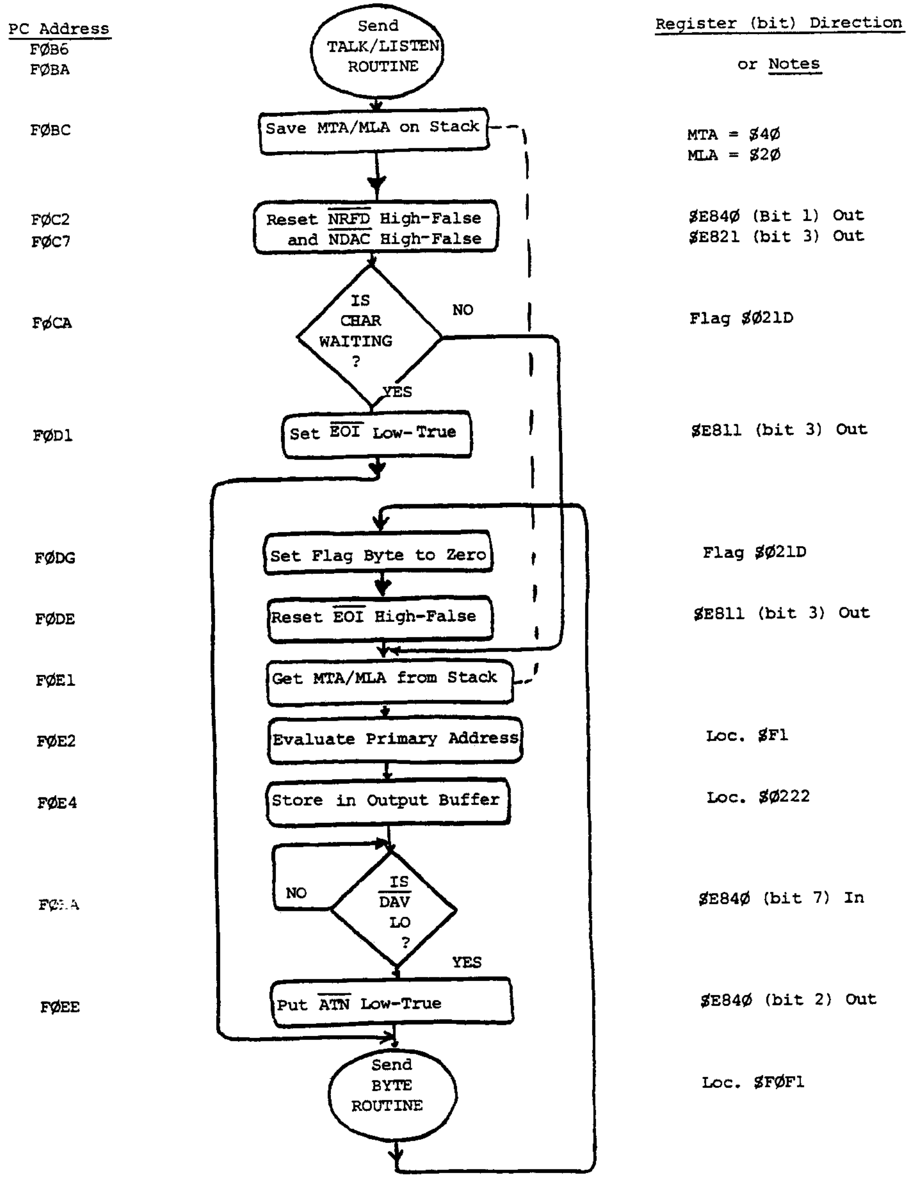
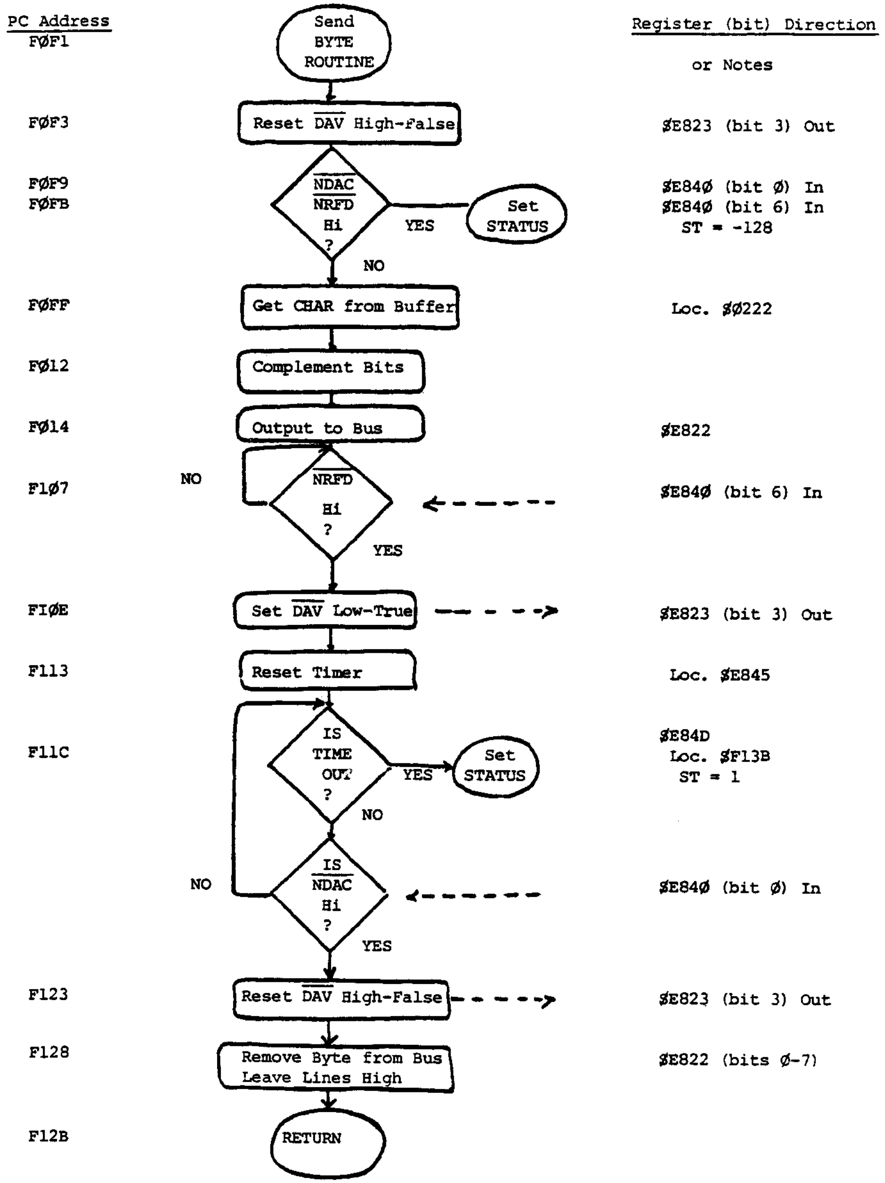


FIG 2
Send Byte (CHAR) to IEEE Bus



Note: Broken lines indicate testing of transfer bus signal logic levels.

FIG 3
Get BYTE (CHAR) from IEEE Bus

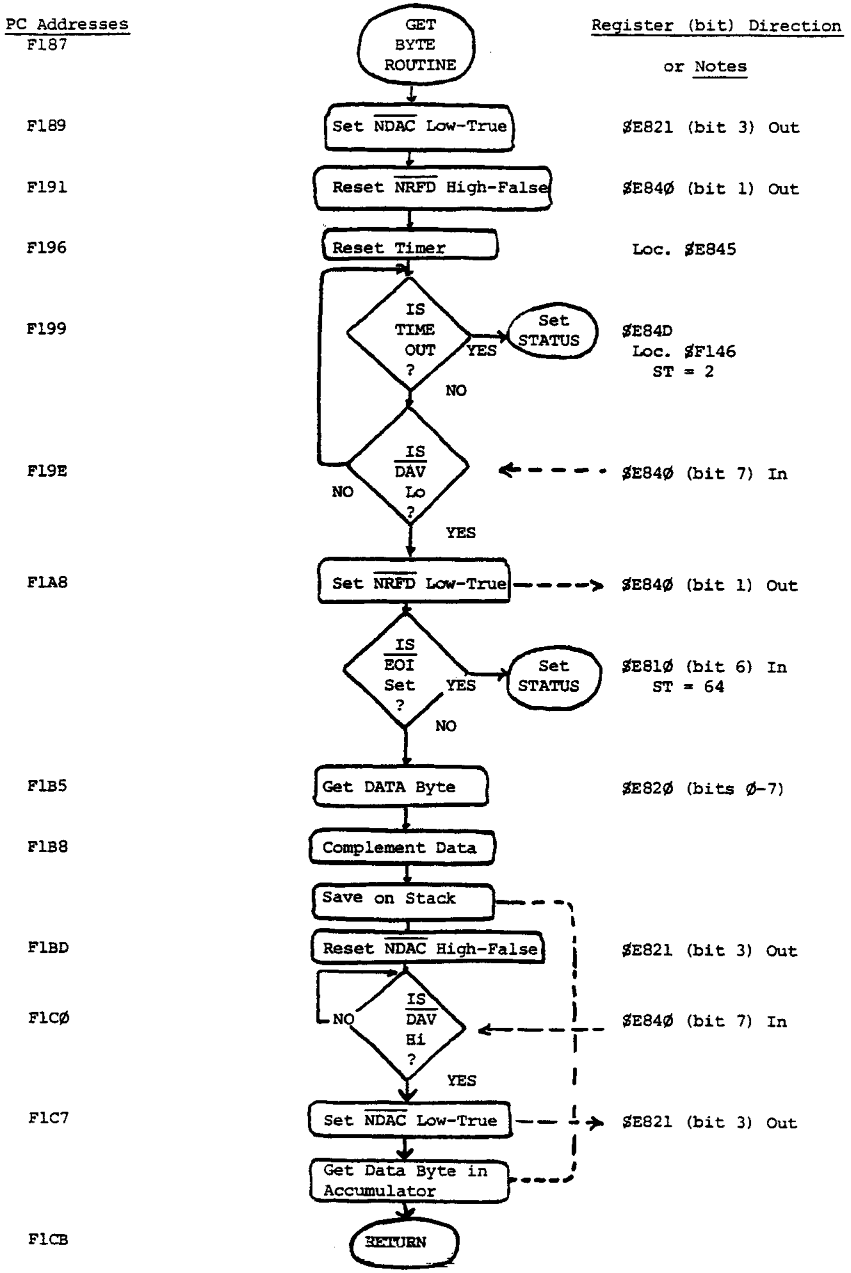


FIG 2a
Handshake Timing Diagram for Send BYTE to IEEE Routines

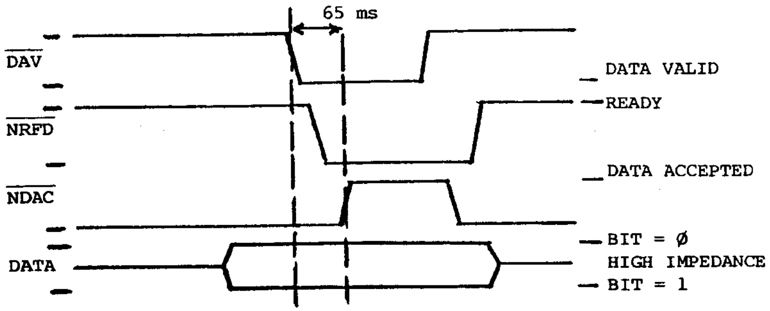
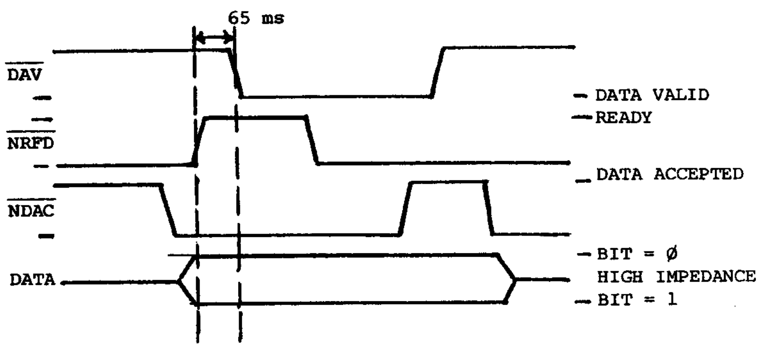


Fig 3a Handshake Timing Diagram for Get BYTE from IEEE Routine



TRANSFER BUS HANDSHAKE PROTOCOLS

IEEE-488 BUS PET MACHINE CODE ROM ROUTINES

```
READY.
FØB6   A9 40      LDA   #$4Ø     ;)@   SEND TALK (MTA) TO IEEE
FØB8   DØ 02      BNE   $FØBC    ;P"
FØBA   A9 2Ø      LDA   #$2Ø     ;)    SEND LISTEN (MLA) TO IEEE
FØBC   48         PHA            ;H      SAVE ADDRESS ON STACK
FØBD   AD 4Ø E8   LDA   $E84Ø    ;-@H
FØCØ   Ø9 Ø2      ORA   #$Ø2     ;)"
FØC2   8D 4Ø E8   STA   $E84Ø    ;-@H    RESET NRFD HIGH-FALSE (READY)
FØC5   A9 3C      LDA   #$3C     ;)<
FØC7   8D 21 E8   STA   $E821    ;-1H    RESET NDAC HIGH-FALSE ALSO
FØCA   2C 1D Ø2   BIT   $Ø21D    ;,="    TEST OUTPUT FLAG BYTE
FØCD   FØ 12      BEQ   $FØE1    ;P2     BRANCH IF NO CHAR WAITING
FØCF   A9 34      LDA   #$34     ;)4     CHARACTER WAITING
FØD1   8D 11 E8   STA   $E811    ;-1H    SET EOI LOW-TRUE
FØD4   2Ø F1 FØ   JSR   $FØF1    ; QP    SEND BYTE )CHAR) TO IEEE
FØD7   A9 ØØ      LDA   #$ØØ     ;)
FØD9   8D 1D Ø2   STA   $Ø21D    ;-="    SET OUTPUT FLAG BYTE TO ZERO
FØDC   A9 3C      LDA   #$3C     ;)<
FØDE   8D 11 E8   STA   $E811    ;-1H    RESET EOI HIGH-FALSE
FØE1   68         PLA            ;H      GET ADDRESS FROM STACK
FØE2   Ø5 F1      ORA   $F1      ;%Q     EVALUATE PRIMARY ADDRESS
FØE4   8D 22 Ø2   STA   $Ø222    ;-""    STORE IN IEEE OUTPUT BUFFER
FØE7   AD 4Ø E8   LDA   $E84Ø    ;-@H    WAIT FOR DAV IN TO GO LOW
FØEA   1Ø FB      BPL   $FØE7    ;Ø[
FØEC   29 FB      AND   #$FB     ;)[     DAV NOW LOW-TRUE
FØEE   8D 4Ø E8   STA   $E84Ø    ;-@H    SET ATN OUT LOW-TRUE
FØF1   A9 3C      LDA   #$3C     ;)<   SEND BYTE (CHAR) TO IEEE
FØF3   8D 23 E8   STA   $E823    ;-#H    RESET DAV OUT HIGH-FALSE
FØF6   AD 4Ø E8   LDA   $E84Ø    ;-@H
FØF9   29 41      AND   #$41     ;)A     TEST FOR BOTH NDAC
FØFB   C9 41      CMP   #$41     ;IA      AND NRFD HIGH-FALSE
FØFD   FØ 43      BEQ   $F142    ;PC     BOTH HIGH - SET STATUS BYTE
FØFF   AD 22 Ø2   LDA   $Ø222    ;-""    GET CHAR FROM OUTPUT BUFFER
F1Ø2   49 FF      EOR   #$FF     ;I      COMPLEMENT BITS FOR BUS
F1Ø4   8D 22 E8   STA   $E822    ;-"H    OUTPUT DATA BYTE TO BUS
F1Ø7   2C 4Ø E8   BIT   $E84Ø    ;,@H    TEST IF NRFD HAS GONE HIGH
F1ØA   5Ø FB      BVC   $F1Ø7    ;P[     IF NOT THEN WAIT
F1ØC   A9 34      LDA   #$34     ;)4     NRFD HIGH-FALSE (READY)
F1ØE   8D 23 E8   STA   $E823    ;-#H    SET DAV OUT LOW-TRUE
F111   A9 FF      LDA   #$FF     ;)      RESET TIMER
F113   8D 45 E8   STA   $E845    ;-EH     FOR 65MS TIMEOUT
F116   AD 4Ø E8   LDA   $E84Ø    ;-@H    WAIT FOR NDAC TO GO HIGH
F119   2C 4D E8   BIT   $E84D    ;,MH    TEST FOR TIMEOUT ON WRITE
F11C   7Ø 1D      BVS   $F13B    ;P=     TIMEOUT ERROR - SET STATUS
F11E   4A         LSR            ;J
F11F   9Ø F5      BCC   $F116    ;ØU     TIME NOT OUT - TEST AGAIN
F121   A9 3C      LDA   #$3C     ;)<     NDAC HIGH (DATA ACCEPTED)
F123   8D 23 E8   STA   $E823    ;-#H    RESET DAV OUT HIGH FALSE
F126   A9 FF      LDA   #$FF     ;)      REMOVE DATA BYTE FROM BUS
F128   8D 22 E8   STA   $E822    ;-"H      LEAVE LINES HIGH
F12B   6Ø         RTS            ;@      RETURN
```

IEEE-488 BUS PET MACHINE CODE ROM ROUTINES

```
IEEE-488 BUS    PET MACHINE CODE ROM ROUTINES

F12C   8D 22 02    STA   $0222    ;-""  SEND SECONDARY ADDRESS
F12F   20 F1 F0    JSR   $F0F1    ; QP   TO IEEE LISTENER DEVICE
F132   AD 40 E8    LDA   $E840    ;-@H   SEND NO SECONDARY ADDRESS
F135   09 04       ORA   #$04     ;)$     NO SEC ADDR SPECIFIED
F137   8D 40 E8    STA   $E840    ;-@H   SET ATN OUT HIGH-FALSE
F13A   60          RTS            ;@     RETURN
F13B   A9 01       LDA   #$01     ;)!    FLAG ST FOR TIMEOUT ON WRITE
F13D   20 E5 FB    JSR   $FBE5    ; E[
F140   D0 DF       BNE   $F121    ;P-
F142   A9 80       LDA   #$80     ;)     FLAG ST FOR DEVICE NOT PRESENT
F144   30 F7       BMI   $F13D    ;0W
F146   A9 02       LDA   #$02     ;)"    FLAG ST FOR TIMEOUT ERROR
F148   20 E5 FB    JSR   $FBE5    ; E[     ON READ
F14B   AD 40 E8    LDA   $E840    ;-@H
F14E   29 FD       AND   #$FD     ;)]
F150   8D 40 E8    STA   $E840    ;-@H   SET NRFD OUT LOW-TRUE
F153   A9 34       LDA   #$34     ;)4
F155   8D 21 E8    STA   $E821    ;-!H   SET NDAC OUT LOW-TRUE
F158   A9 0D       LDA   #$0D     ;)-
F15A   60          RTS            ;@     RETURN
F15B   8D 22 02    STA   $0222    ;-""  SEND SECONDARY ADDRESS TO
F15E   20 F1 F0    JSR   $F0F1    ; QP   IEEE TALKER DEVICE
F161   20 4B F1    JSR   $F14B    ; KQ   SET NRFD AND NDAC LOW
F164   4C 32 F1    JMP   $F132    ;L2Q   SEND NO SECONDARY ADDRESS
F167   2C 1D 02    BIT   $021D    ;,="   TEST OUTPUT FLAG BYTE
F16A   30 05       BMI   $F171    ;0%    BRANCH IF CHAR WAITING
F16C   CE 1D 02    DEC   $021D    ;N="   NO CHAR - RESET BUFFER
F16F   D0 05       BNE   $F176    ;P%
F171   48          PHA            ;H     SAVE ACC ON STACK
F172   20 F1 F0    JSR   $F0F1    ; QP   SEND DEFERRED CHAR TO IEEE
F175   68          PLA            ;H     GET CURRENT CHAR IN ACC
F176   8D 22 02    STA   $0222    ;-""   STORE IN OUTPUT BUFFER
F179   60          RTS            ;@     RETURN
F17A   A9 5F       LDA   #$5F     ;)-   SEND UNTALK TO IEEE
F17C   D0 02       BNE   $F180    ;P"    UNTALK ALL DEVICES ON BUS
F17E   A9 3F       LDA   #$3F     ;)?   SEND UNLISTEN TO IEEE
F180   85 F1       STA   $F1      ;%Q    UNLISTEN ALL DEVICES
F182   20 BC F0    JSR   $F0BC    ; <P    SEND TO IEEE BUS
F185   D0 AB       BNE   $F132    ;P*    SET ATN OUT HIGH - FALSE
```

IEEE-488 BUS PET MACHINE CODE ROM ROUTINES

```
READY.
F187   A9 34       LDA   #$34     ;)4    INPUT BYTE (CHAR) FROM IEEE
F189   8D 21 E8    STA   $E821    ;-!H     SET NDAC OUT LOW-TRUE
F18C   AD 40 E8    LDA   $E840    ;-@H
F18F   09 02       ORA   #$02     ;)"
F191   8D 40 E8    STA   $E840    ;-@H     RESET NRFD OUT HIGH-FALSE
F194   A9 FF       LDA   #$FF     ;)
F196   8D 45 E8    STA   $E845    ;-EH     RESET TIMER FOR TIMEOUT
F199   2C 4D E8    BIT   $E84D    ;,MH     TEST FOR TIMEOUT ON READ
F19C   70 A8       BVS   $F146    ;P(      BRANCH IF TIMEOUT ERROR
F19E   2C 40 E8    BIT   $E840    ;,@H     TEST IF DAV HAS GONE LOW
F1A1   30 F6       BMI   $F199    ;0V      IF NOT TEST AGAIN
F1A3   AD 40 E8    LDA   $E840    ;-@H     DAV NOW LOW (DATA VALID)
F1A6   29 FD       AND   #$FD     ;)]
F1A8   8D 40 E8    STA   $E840    ;-@H     SET NRFD LOW-TRUE
F1AB   2C 10 E8    BIT   $E810    ;,0H     TEST IF EOI IS SET
F1AE   70 05       BVS   $F1B5    ;P%      BRANCH ON EOI NOT SET
F1B0   A9 40       LDA   #$40     ;)@      EOI NOW SET LOW-TRUE
F1B2   20 E5 FB    JSR   $FBE5    ; E[     SET STATUS 'EOI'
F1B5   AD 20 E8    LDA   $E820    ;- H     INPUT DATA BYTE FROM IEEE
F1B8   49 FF       EOR   #$FF     ;I       COMPLEMENT DATA (ACTIVE HIGH)
F1BA   48          PHA            ;H       SAVE DATA BYTE ON STACK
F1BB   A9 3C       LDA   #$3C     ;)<
F1BD   8D 21 E8    STA   $E821    ;-!H     RESET NDAC HIGH (DATA ACCEPTED)
F1C0   2C 40 E8    BIT   $E840    ;,@H     TEST IF DAV HAS GONE HIGH
F1C3   10 FB       BPL   $F1C0    ;0[      BRANCH ON DAV STILL LOW
F1C5   A9 34       LDA   #$34     ;)4      DAV NOW HIGH-FALSE
F1C7   8D 21 E8    STA   $E821    ;-!H     SET NDAC OUT LOW-TRUE
F1CA   68          PLA            ;H       GET DATA BYTE IN ACCUMULATOR
F1CB   60          RTS            ;@       RETURN
```

Education

The Missing Link

(An article by Nick Green, Commodore's Special Projects Manager)

At a Commodore Education conference in London, teachers insisted on a more systematic approach to the provision of C.A.L material for the curriculum.

In total around the country the past twelve months have seen the development of well over 1000 pieces of teaching material in the "Teachers' Aid" category.

The need is to co-ordinate a similar software writing effort in such a way as large areas of the curriculm are covered. The method is for Subject Matter Experts e.g. people who have written text books, or groups of concerned teachers, to produce a breakdown into topics of the particular areas they wish to tackle. These topic lists should be used to indicate how many programs or sets of programs have to be written to cover that area of the curriculum.

If Subject Matter Experts or Subject Matter Expert Groups have any difficulties in proposing topic lists they should consult examination syllabi or some of the revision notes that are published by many companies for particular subjects.

If there is a group of teachers who want to tackle a particular subject area we will be happy to publish their topic lists and details of which topics are having programs written for them and which topics require teacher-programmers. Send your topic lists, with notes on which topics are being tackled by your group, to Nick Green or Jean Frost at Slough and we'll publish them in future issues.

reen, Special Projects Manager, in cheerful mood

Low Priced Software

Getting hold of software at a low price is very difficult in today's market place. This situation is due to the normal commercial pressures of cost recovery and profit generation, however, when the former is low and the latter is low then prices can reflect it. This is the situation at Qwerty Computer Services; we have refrained from heavy advertising and kept other costs as low as possible — the result is very competitive priced software. No programs in our present catalogue cost more than £1.25 (plus a little contribution towards post and packing) and they all come on cassette - not listings. Many people have bought the software and **no-one** has complained at their value - we do not provide documentation - wherever possible it is incorporated into the program. The range of software is fairly good, interactive games, business simulation, utilities and novelties. For educational use we have various programs which have been tried out in the classroom.

Remedial:

A 3 program suite which allows the teacher to drill students in spelling, word formation and vocal presentation. The program is for setting up data files of words, sentences or phrases which will be presented to the student by the Remedial Run program. This second program does the actual testing/drilling e.g. Prints word etc to screen, generates signals for teacher vocalisation, accepts pupils input and vocalisations, prepares an analysis file for the last program: Analysis is provided in the form of error display -the word is displayed then each letter attempted incorrectly is shown.

Payroll Simulation and Business Simulation:

Shows the effect of taxation, insurance, overtime etc. upon pay. Run a company -decide number of men, raw material, advertising, premises etc, results shown for each year's trading - interactive.

Name/Grade Sort:

Sort any size class plus marks for any number of subjects into alphabetical order or class order **per subject** - no more than one minute for a batch size of 500 pupils - whole year groups.

Junior and Infants:

Word-build helps pupils increase word recognition. Good graphics provide stimulus for correct answers.

Also available is the following hardware.

Soundbox £15, T.V./Video Interface £31.50, Combined Soundbox/T.V. Video Interface £45, Switch Unit £11 plus £5 per switch (extra £1.50 for mains switching), Light Pen £15, Crash Restorer £7, A/d Converter 12 bit Resolution £55, plus various other bits and bobs.

I must stop here otherwise I will take up the whole publication. One last thing - we pay 15% royalty and need more programs - send them to the address below and help us to provide a greater range of software at realistic prices. For current catalogue and price list send s.a.e. to:

Qwerty Computer Services,
20 Worcester Road,
Newton Hall,
Durham DH1 5PZ.

Manchester Education User Group

Following the introduction of Pet microcomputers into Manchester schools during the past year the work of co-ordinating their uses and impact has started.

In order to provide a central sevice for teachers in all areas of microcomputing using the PET a user group has been established.

The user group will provide a forum for developing the use of microcomputers in education and provide a software support service.

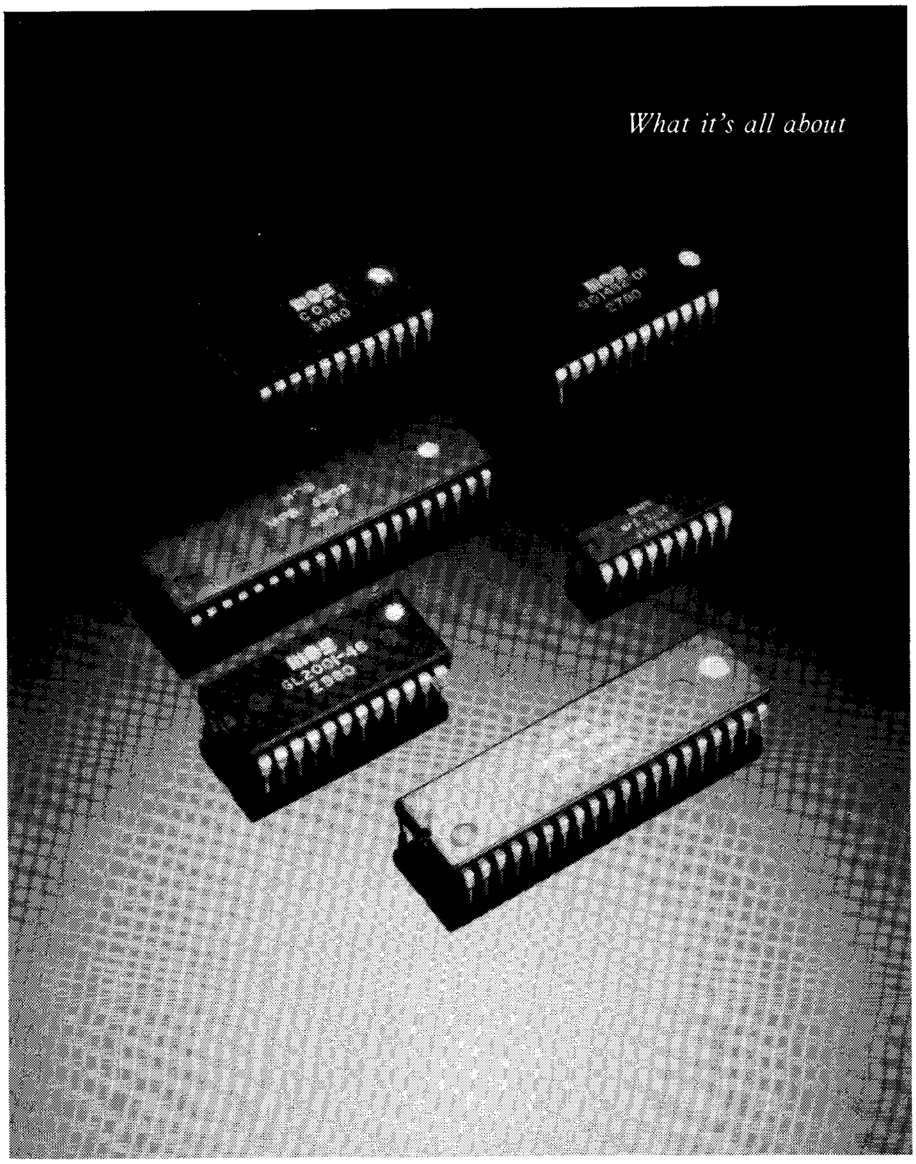
A recent survey of school's software showed an abundance of programs in mathematics and science but a shortage in humanities and languages. The user group is hoping to find people with interests in these areas and help them to produce suitable software for use in schools. The Survery results have been compiled into a reference list for schools, a draft list of the user groups library and to inform teachers of contact within the authority.

It is hoped to publish termly up dates of this list with the microcomputer bulletin of local information, news, recent developments and programming hints.

The appointment of a full time co-ordinator for microcomputing has meant an expansion in the services to be available to local schools. The software advice service for the curriculumn is being expanded to include administration programming and hardware advice. At the moment the area of control technology is generally left untouched in most schools. It is hoped in the future to be able to provide schools with the information and help in using their PET to control experiments and other peripherals within the class. The establishment of a control technology group may then become a neccessity.

Data Based Administration Programs

Already a few schools are running Data Based Administration programs. These are mainly concerned with the saving of examination results, student course information and the printing of class lists. The systems in use are being run by home produced software compiled by the



What it's all about

individual schools and tailored to their requirements. The portability of such software to other schools needs is being investigated as well as the extension to include information needed for the D.E.S. form 7.

The final, and most important area, of work in the authority is the provision of service training courses. It is hoped during the next year to have service courses in all area of educational computing. Courses are now being planned for:- using the PET in administration; using the PET in the classroom; programming in BASIC and programming in assembler.

The user group would be pleased to hear from other educational user groups with a view to future exchanges of ideas and help in educational developments of microcomputing. The work in Manchester is being co-ordinated by:- Mr P. Murphey, Teachers Centre, Barlow Moor Road, Manchester, Mr A. Goodall, Xaverian College, Lower Park Road, Manchester 14

A. Goodall, Xaverian College, Manchester

High Resolution Graphics for the PET

The MTU high resolution graphics package distributed in the UK by IJJ Design Limited is a simple and inexpensive addition to the PET which allows the full resolving power of the monitor to be realised for plans, drawings, graphs, forms, games and the like, with complete software control over the display, involving the superimposition of text and keyboard graphics symbols.

The system consists of 8K of additional memory which contains the data for a 320 x 200 matrix of dots on the screen, and comes in two forms, mounted internally or externally.

The external version resides in a box alongside the PET, which also contains slots for two other MTU boards, such as the PROM programming board and the memory expansion board. It also has a socket which provides a composite video signal suitable for driving a video monitor dedicated exclusively to graphics display.

The internal version provides a convenient intergrated system within the PET, with additional control over the display to simplify the use of the same screen for both purposes. It also has five ROM sockets which provide a home for ROMs such as Toolkits and Word Processors which occupy the same addresses as the graphics memory. These are addressed whenever the display is in the normal mode, the graphics taking their place in graph mode.

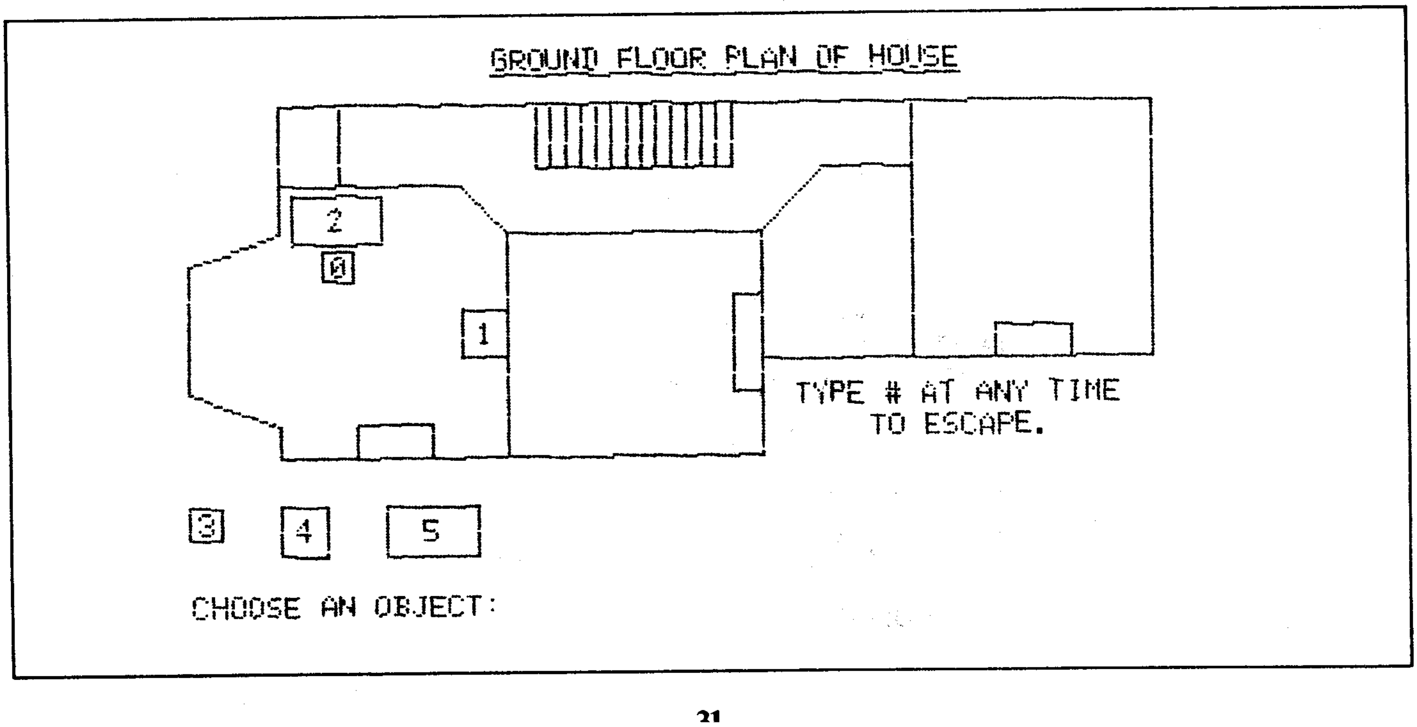
Advanced software is available which gives powerful control over the system through the use of 19 BASIC keywords, which can be incorporated into the programme in just the same way as any other command. The advanced software, called PETGRAPH, occupies 4K of RAM, and includes command for the control of the screen, for drawing dots, lines and dotted lines using xy coordinates, for superimposing text and for declaring, moving and storing chosen objects on the screen.

A Typical Application Example

A typical application example (which is included with the software) draws a ground-plan of house and a selection of numbered shapes representing furniture to be fitted into the plan. The user selects a shape number and a flashing dot at the bottom left-hand corner of the shape acts as a cursor, which can then be repositioned anywhere on the plan by using the number keys in the usual way. When the desired position is reached the space bar is pressed and the shape then glides smoothly across the screen to its chosen destination.

A new shape then takes its place in the line-up of shapes for selection and the process may be repeated. The shapes may be moved as many times as desired, until the layout meets with the approval of the user. Having achieved a plan using the above example programme, the Print Graph programme may be loaded to print out the plan on a Commodore 3022 tractor printer.

The user of this graphics system makes a valuable and easily used addition to the PET which greatly increases its potential as an educaional, research or business machine. In particular, its utility in the areas of visual aid and computer assisted learning will be apparent immediately to teachers and lecturers.



GROUND FLOOR PLAN OF HOUSE

TYPE # AT ANY TIME
TO ESCAPE.

CHOOSE AN OBJECT: