PERSONAL COMPUTER WORLD

50p U.S.$2.00

NOVEMBER 1978

INTERATIONAL CHESS MASTER DAVID LEVY ON OUR MICROCHESS CHAMPIONSHIP

Europe's first magazine for personal computers for home and business use

The Micro Muse

Learning Geography with a Computer

Interfacing a Pet to an X·Y Plotter

Benchmarks Again



MIKE JOHNSON: Microchess Champion

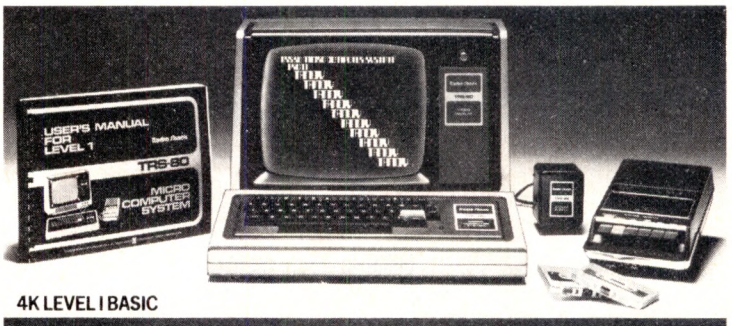
The TANDY TRS 80 the ultimate in personal computers -right off the shelf.

Five complete low-cost systems incorporating different combinations of RAM (4K and 16K) and ROM (Level 1 and Level 2 BASIC) Designed and manufactured by Tandy in the USA, 100% wired, tested and ready to go.

4K LEVEL I BASIC

Low cost breakthrough!
*TRS-80 Micro-computer
*12" video display
*Easy 'typewriter' keyboard.
*Power transformer
*Cassette tape recorder
*4K RAM, Level-1 BASIC
*232 page manual
*2 game cassettes
26-1001

£499

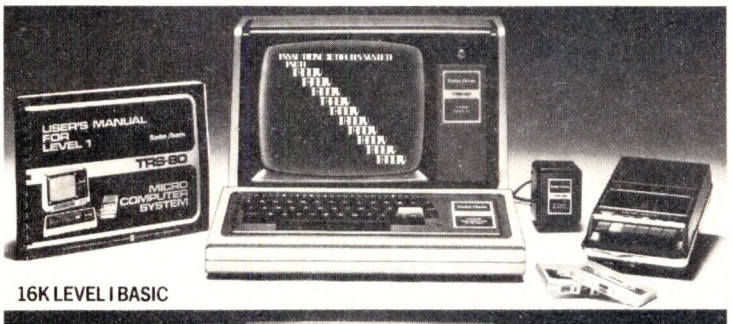


4K LEVEL I BASIC

16K LEVEL I BASIC

The same hardware as the 4K system but with 16K RAM it gives greater data storage and programming capacity. 26-1003

£728

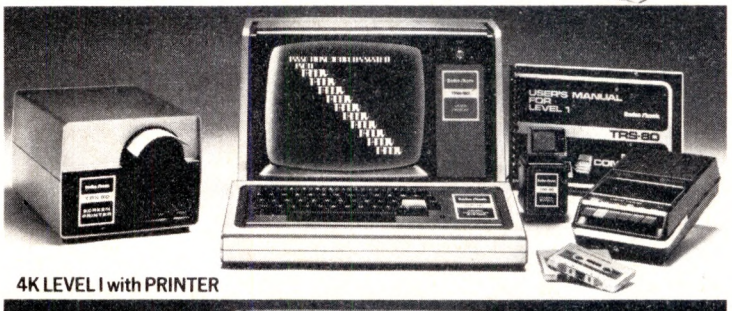


16K LEVEL I BASIC

4K LEVEL I with PRINTER

Our basic system but with a hard-copy screen printer. At the touch of a button the screen display is printed out on electrostatic paper as a permanent visual record. (26-1001/26-1151)

£978

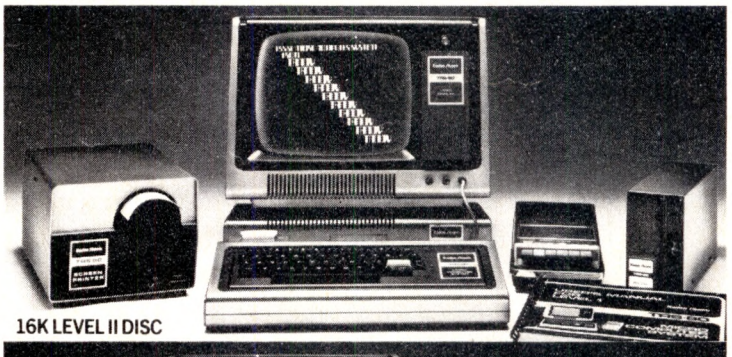


4K LEVEL I with PRINTER

16K LEVEL II DISC

Professional 16K RAM operation with mini-disc, screen printer and expansion interface. (26-1006/26-1151/26-1140/26-1160)

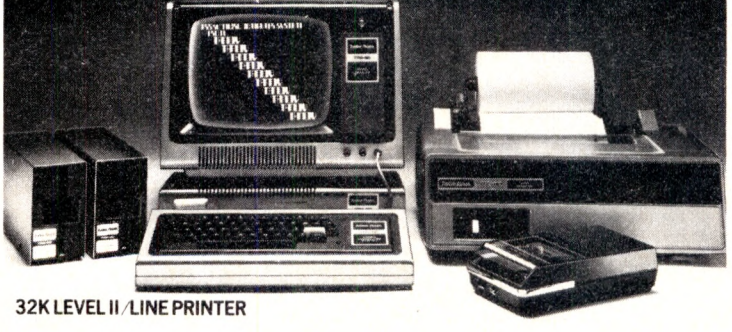
£1914


16K LEVEL II DISC

32K LEVEL II/LINE PRINTER

This is the ultimate! 32K RAM, 2 mini-discs, line printer, expansion interface, 16K memory option. A full business system.
(26-1006/26-1160/26-1161/26-1150/26-1140/26-1101).

£3058


32K LEVEL II/LINE PRINTER

See the TRS-80 micro-computer breakthrough at most Tandy stores and dealerships or write now for further data to:-

Tandy Corporation, Computer Division, Bilston Road, Wednesbury, Staffordshire WS10 7JN.

TANDY

The biggest name in little computers.

PERSONAL COMPUTER WORLD

Europe's first magazine for personal computers for home and business use

UK 50p
US $2.00

Vol 1, No 7
November 1978

CONTENTS

PUBLISHER'S LETTER 5

EDITORIAL NOTE 5

LETTERS 7

TIDBITS 9

THE MICRO MUSE *Eric Finlay*
Poetry and the computer 13

PCW MICROCOMPUTER CHESS CHAMPIONSHIP *David Levy*
Chess and the computer. A closely fought finish to three days of excitement and entertainment 15

AN AGILE ALGORITHM *R.E.L. Ferguson*
Square root by addition 19

PRIME NUMBERS COMPETITION *Sheridan Williams*
Results of efforts by programmers in their prime 21

A WAY WITH WAVES *Gordon Bell*
Feed this to your Pet 25

ALIGNING THE DECIMAL POINT IN BASIC *Sidney A. Leleux* 26

GETTING IT TOGETHER *Mike Banahan*
Continuation of listing for the "own assembler" series. 27

USING A SMALL COMPUTER *P. M. Steele*
A Consultant considers mailing lists 31

HOW TO CONTROL YOUR BASIC FUNCTIONS
Christopher Smith.
We did'nt select the title: the author did 33

HOW TO ENTER COMPUTER PROGRAMMING *George Penney*
For our many readers in School or considering a new career 35

LEARNING GEOGRAPHY WITH A COMPUTER
John D. Lee and Timothy D. Lee
Computer aided instruction. Relief for the teacher, fun for the pupil 37

PCW OPEN PAGE *Mike Lord*
The inimitable forum for the amateur 39

OUR SHOW Scenes from a great occasion 41

BOOK REVIEW *Michael James* 42

THE POWER OF POSITIVE TABLING *R. W. Davy*
A powerful programming aid 45

PLAN FOR A PLOTTER *R. I. Elliott*
Interfacing a Pet to an X-Y Plotter 49

BENCHMARKS AGAIN AND CONNECTIONS *John Cole*
This illustrates **PCW** power: its value equals pages in other magazines 52

BUZZWORDS *Peter Reynolds*
Helps you to swim in heavy C'zz 54

PERSONAL EXPERIENCE *G. D. Hayes*
A member of the ACC and his machine 57

KEEPING AN ORDERLY HOUSE *A. Clements*
Monitors for Microprocessors 60

STAYING IN SHAPE *Michael James*
An influential style: Structured programming 66

ISSN:0142 0232

Editorial and Advertising Office:
62A Westbourne Grove, London W2
Phone: 01-229 5599 (Publisher)
01-727 8758 (Editorial and Advertising)

Publisher:
A. Zgorelec
Editor:
Meyer N. Solomon
Editorial Assistant:
Roger C. Wilkins
Policy Advisor:
Peter Crofton-Sleigh, FRAS

Consultants:
John Coll, Mike Dennis, Neil Harrison, Charles Sweeten, Patrick Sutton, Michael James, R.W. Davy, David Hebditch, Sheridan Williams, Stephen Castell
Art: **Sauveur Laurent Sant, Kathryn Hamme**
Secretarial:
Vanessa Blackburn Kiddle

Layout Consultants:
D. Norris, T. Gabos
PCW Photography: **Yoshi Imamura, Peter McGee**
Typesetting & Artwork: **Gilfillan**

Cover photo: **Yoshi Imamura**

CONTRIBUTORS:

We welcome interesting articles written simply and clearly. You need not be a specialist to write for us. MS should not be more than 3000 words long, lines double spaced, with wide margins. Line drawings and photographs wherever possible. Enclose a stamped self-addressed envelope if you would like your article returned.

Manufacturers, suppliers and dealers are welcome to contribute technical articles, and send product information, but we are pledged to an independent viewpoint and will publish evaluations and reasoned criticism or praise, space permitting. Naturally there will be right of reply. Views expressed in articles are not necessarily those of Personal Computer World.

We may make arrangements to offer our readers products at special prices, for a limited period, in line with the policy outlined above.

Published monthly by Intra Press, 62A Westbourne Grove, London W2. Phone: 01-229 5599. Contents fully protected by copyright. All rights reserved. Subscription rates: Britain £8 for 12 issues. Prices include postage. USA – $10 for six issues, $20 for 12 issues. Continent and elsewhere: £9.80 for twelve issues. Prices include postage.
Printed by Carlisle Web Offset, 55 Conduit Street, Newtown Trading Estate, Carlisle CA2 7NR. Sole UK Distributors: Seymour Press Ltd., 334 Brixton Road, London S.W.9., England. Distribution to specialist shops by Intra Press.

Room P.W.29
313 Kingston Road, Ilford,
Essex, IG1 1PJ, England
01-553 1001

L.P. ENTERPRISES

From the representatives in Europe . . . for America's leading Micro-computer magazines and books, for the hobbyist, educationist and professional alike, we bring you a little light browsing!

Reading maketh a full man . . . . Francis Bacon (1561–1626)

Tick or indicate quantity ordered

From Adam Osborne Associates	
INTRODUCTION TO MICROCOMPUTERS	
Volume 0: The Beginners Book	£5.95
Volume 1: Basic Concepts	£5.95
Volume 2: Some Real Products (Revised Late 1977)	£11.95
6800 Programming for Logic Design	£5.95
8080 Programming for Logic Design	£5.95
Z80 Programming for Logic Design	£5.95
8080A/8085 Assembly Language Programming	£6.95
6800 Assembler Language Programming	£6.95
Some Common BASIC Programs	£5.95
BUSINESS PROGRAMS IN BASIC	
Payroll with Costing Accounting	£9.95
Accounts Payable & Accounts Receivable	£9.95
From Scelbi Computer Consulting Inc.	
6800 Software Gourmet Guide & Cookbook	£7.95
8080 Software Gourmet Guide & Cookbook	£7.95
8080 Programmers Pocket Guide	£2.25
8080 Hex Code Card	£2.25
8080 Octal Code Card	£2.25
8080 Guide and One 8080 Code Card	£4.20
8080 Guide and Both Code Cards	£6.00
Understanding Microcomputers & Small Computer Systems	£7.95
SCELBI 'BYTE' Primer	£9.95
8080 Standard Assembler (In Book Format)	£15.95
8080 Standard Editor (In Book Format)	£9.95
8080 Standard Monitor (In Book Format)	£7.95
8080 Galaxy Game (In Book Format)	£9.95
From Peoples Computer Company	
Reference Books of Personal & Home Computing	£4.95
What to Do After You Hit Return	£7.00
Dr. Dobbs Journal Volume 1	£10.00
From Kilobaud/73 Magazine Inc.	
Hobby Computers Are Here	£3.95
New Hobby Computers	£3.95
From Dymax Inc.	
Instant BASIC by Jerald R.Brown	£4.95
Your Home Computer by James White	£4.95
My Computer Like Me . . . When I Speak BASIC by Bob Albrecht	£1.65
Games with a Pocket Calculator by Thiagarajan & Stilovitch	£1.75
Games, Tricks and Puzzles for a Hand Calculator by W.Judd	£2.49
From BYTE Publications Inc.	
Paperbytes:	
Tiny Assembler for 6800 Systems	£5.75
Bar Code Loader for 6800, 8080, Z80 & 6502 Micros	£1.75
Best of BYTE Volume 1	£8.95

Tick or indicate quantity ordered.

	Price UK	Price Overseas If Different
From Creative Computing Press		
Best of Creative Computing Volume 1	£6.95	
Best of Creative Computing Volume 2	£6.95	
BASIC Computer Games (Revised & Reprinted Feb. 78)	£5.50	
The Colossal Computer Cartoon Book	£3.95	
Computer-Rage (A new Board Game)	£6.95	
Artist and Computer	£3.95	
From Everyone Else		
Magazine Storage boxes (holds 12 minimum)	£1.25	
Sybex: Introduction to Personal and Business Computing	£4.95	
Sybex: Microprocessors from Chips to Systems by R.Zacs	£7.95	
Sybex: Microprocessors Interfacing Techniques by R.Zacs	£7.95	
Dilithium: Home Computers Volume 1: Hardware	£6.50	
Dilithium: Home Computers Volume 2: Software	£5.95	
Getting involved with your Own Computer	£4.75	
TV Typewriter Cookbook by Don Lancaster	£7.50	
TTL Cookbook	£7.50	
CMOS Cookbook	£7.95	
IC Timer Cookbook	£7.50	
IC OP–AMP Cookbook	£9.50	
RTL Cookbook	£4.25	
Z-80 Microprocessor Handbook	£7.50	
Computer Programs that work (In BASIC)	£2.55	
From Basic Software Library (from Scientific Research Instruments)		
Vol 1: Business and Personal Booking Programs	£17.50	
Vol 2: Maths and Engineering Programs	£17.50	
Vol 3: Advanced Business Programs	£26.95	
Vol 4: General Purpose Programs	£7.95	
Vol 5: Experimenters Programs (General Purpose)	£7.95	
Vol 6: General Ledger Program	£32.50	
Vol 7: Professional Programs	£26.95	
Magazines: Back Issues		
Personal Computing	£1.75	
Interface Age	£2.25	
Dr. Dobbs Journal	£1.75	
Computer Music Journal	£2.50	
Peoples Computers	£1.75	
BYTE	£2.25	
Creative Computing	£1.75	
Calculators & Computers	£1.75	
ROM	£1.75	
Kilobaud	£2.25	
73	£2.25	
6502 Journal	£1.75	
MAGAZINES: Subscriptions		
Personal Computing (Twelve Issues Yearly)	£16.00	£17.00
Interface Age (Twelve Issues Yearly)	£20.00	£20.50
Dr. Dobbs Journal (Ten Issues Yearly)	£13.00	£13.50
Computer Music Journal (Four Issues Yearly)	£8.50	£9.00
Peoples Computers (Six Issues Yearly)	£8.00	£8.50
Kilobaud (Twelve Issues Yearly)	£20.00	£21.00
BYTE (12 issues yearly) via U.K.	£21.00	
Creative Computing (Twelve issues yearly)	£16.00	£17.00
Calculators & Computers (Seven Issues Yearly)	£10.00	£10.50
73 (Twelve Issues Yearly)	£20.00	£21.00
6502 Journal (Six issues Yearly)	£9.00	£9.50

THIS LIST CANCELS ALL PREVIOUS PRICE LISTS: EFFECTIVE AUGUST 1978

PCW/878/VC

HOW TO ORDER

Please note our prices include postage and packing, but not insurance, if wanted add 12p for every £10. of **books** ordered. Make cheques, PO's etc. payable to:–

L.P. Enterprises.

CREDIT CARDS accepted
BARCLAYCARD VISA / ACCESS
DINERS CLUB / AMERICAN EXPRESS

Phone: 01-553 1001 for Credit Card orders (24–hr service)

Send to address above
Indicate Payment Method:

All Orders must be Prepaid

Total Enclosed £ . . . . . .

. . . . . My cheque, P.O., I.M.O. is enclosed in **Sterling** on **U.K. Bank**

. . . . . Charge to Barclaycard/Visa/Access/Diners/American Express

Credit Card No . . . . . Expiry Date . . . . . .

Name . . . . . .

Address . . . . . .

. . . . . POSTCODE . . . . . .

Signature . . . . . .

All publications are published in U.S.A. and shipped air-freight by **L.P. Enterprises.** In unusual cases, processing may exceed 30 days.

U.K. Subscriptions start within 3 weeks.

TRADE ENQUIRIES WELCOME

Publisher's Letter

Dear Reader,

Our Show. There has been nothing like it in Europe. We had a record attendance, and exhibitors reported record business and enquiries*. Most people came knowing what to look for and what they wanted. The whole atmosphere was one of friendliness and engagement. It was truly a personal computer world, from the video games on display, through the computers for the home, for the small business, for education, to the Chess Championship which had an exciting ending. I presented a cheque for £200 to the winner, Mike Johnson, on behalf of **PCW.**

As you know, Interbuild Exhibitions helped us to organise our Show, and did a superb job. I would like to thank Will Martin and Peter Tuke of Interbuild, as well as all their staff and our consultants.

An amusing thing happened during the Show. Companies don't automatically exhibit. There are other shows around, one of which occurred close enough in time for some companies to decline to exhibit at ours. A couple of such businesses sent people along to see how we were doing. They took one look at the busy stands, and straight away put their names down – one year in advance – for the next Show.

Advertisers are beginning to sit up and take notice: the Show has proved that **PCW** readers know what they want, shop carefully and not frivolously and are willing to spend considerable amounts on systems for themselves or for the companies or institutions they represent.

See you at next year's Show.

**Among the exhibitors was Peter Jennings from America, who brought over his Personal Software from the States. He said, "This has been the best and friendliest Show I have ever experienced.*



The Computer Magazine You Can Read!

You don't have to be a programmer or computer scientist to read Personal Computing. It's the magazine that tells you how to get started in computing. How to have fun. And how to use the computer for your own personal and business applications.

Each month Personal Computing is loaded with practical, fun articles designed to help you get the most out of your computer. We're the magazine with style, color and practicality. Topics covered in recent issues include: Getting Into Computer Games, Using the Computer to Manage a Drug Store, Using Computers in Schools, Learning to Program in Three Easy Lessons, Ten Easy Steps to Become a Computer Hobbyist, How to Set Up a Word Processing System, The Future of Robots, A Computer That Speaks English, Intelligent Video Games, How to Profit from Your Computer Hobby...and much, much more.

Open your eyes, put on your thinking cap and subscribe to PERSONAL COMPUTING!

NAME

ADDRESS

CITY POST CODE

UK
☐ 1 year (12 issues) £16.00

OVERSEAS
☐ 1 year (12 issues) £17.00
(Please remit payment in UK funds on UK Bank)
☐ Cheque enclosed

Charge my:
☐ Barclaycard
☐ Access
☐ American Express
☐ Diners Club
Account #
Card Expiry Date
Signature

BACK ISSUES £1.75/COPY
(Payment must accompany order).

Trade Enquiries Welcome

Mail to:
L. P. ENTERPRISES
313 Kingston Road,
Ilford, Essex, IG1 1PJ England

Editorial

Small Business

An area in which the small computer will have a great impact is the small business. From our very first issue we recognised this and floated the idea of CASBA – the Computer Association for Small Business Applications. And at our Show, finally, enough people put their names down for CASBA to come into being.

CASBA is for both the beginner to small business computing and for those businesses that already have some experience in the use of the small computer. Members will give the benefit of their experience to the beginner; useful software will be exchanged; as will ideas to improve efficiency; not least of all, because members are actually using or intend buying systems, they can in a united way put forward suggestions to suppliers and have their interests considered seriously.

Joining CASBA is very simple. All you need do is write to us and we'll add your name to the list. It's a brilliant way to keep in touch with developments and ease the pains of entry into small business computing.

New Consultant

We are pleased to welcome *Dr. Stephen Castell* as a Consultant to Personal Computer World.

Dr. Castell has been a computer consultant with Touche Ross & Co. and until recently was Group Management Services Manager of the international banking group Bremar Holdings Ltd. He is now an independent management and financial consultant in information technology with interests in projects ranging from professional mainstream computing to microprocessors, personal computing and viewdata, as well as marketing and venture capital, with in particular an emphasis on new-technology business and small-business information systems.

Dr. Castell has in the past written technically, widely and imaginatively in the industrial research, commercial computing and technology spheres, having coined such concepts as the "probit", a mixed processing/data-storage architectural element and "softwiring" or high-level language microprogram-burning. He is currently authoring a series of "Quirkions" (= "Quirky Questions") in the trade weekly Computing, dealing in characteristically idiosyncratic fashion with many fringe and futuristic areas of information technology.

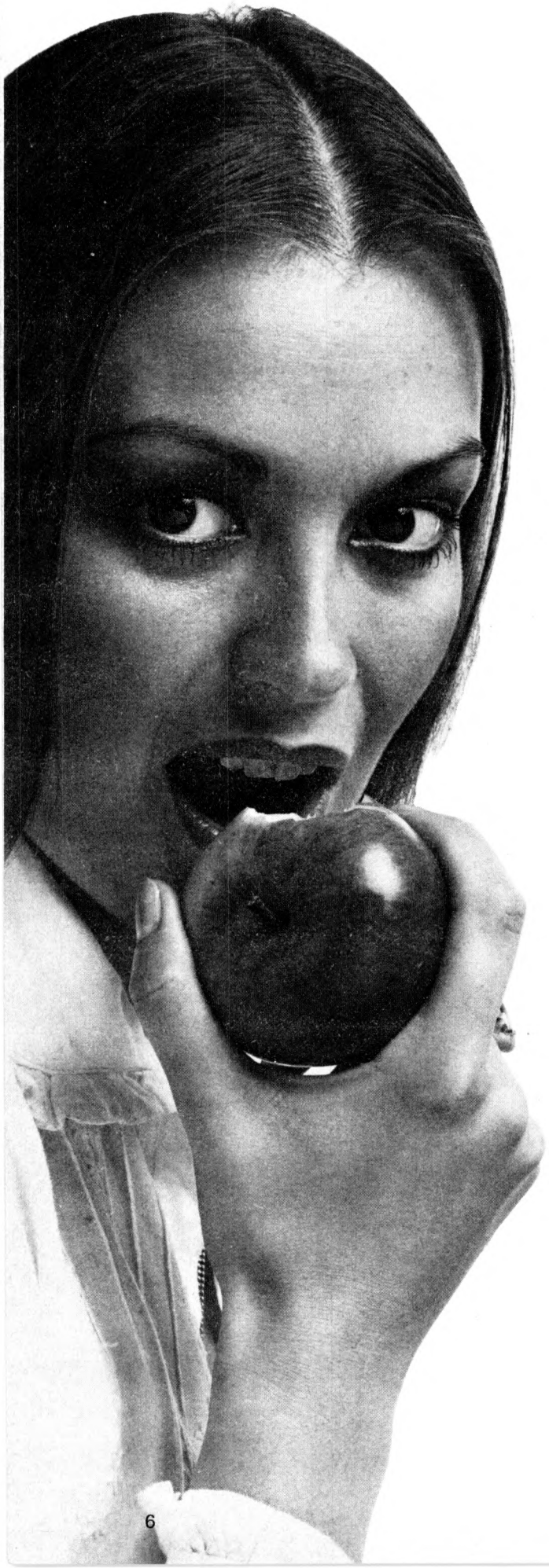
We expect Dr. Castell to fill a much-needed area of contribution to our magazine in covering aspects of microcomputing systems/financing in small business applications.

RENUMBER YOUR BASIC PROGRAMS

This program will completely renumber your old or new SWTPC 8K V2.0 BASIC programs. Comes complete with cassette and fully Documented Source Listing. Will work with cassette or disk versions of BASIC – £30.
COMPUTAWARE, 479 WELLINGBOROUGH RD., NORTHAMPTON. 714821

FOR SALE

Floppy Disc Controller for SWTPC-6800 with Driver Software in Prom **£125**
KSR-35 Teletype 20mA Loop ASC11 **£150**
Other items and software – *write or phone for list.*
ERIS Braeside, Clarach Road, Borth, Dyfed.
Telephone – 097081 267.



I BYTE

BUT COMPUTERS DON'T

Come and see for yourself at

the BYTE SHOP Ltd

Stockists of the largest range of micro computers in the U.K.

Take the opportunity to experiment with and get to know any of the vast range of micro computers always in stock at The Byte Shop.

Whether you want a micro computer for your home, your business, for industry, for education - or if you'd just like to find out which model you get on with best - you'll find a visit to The Byte Shop a new and invaluable experience.

Call in at The Byte Shop any time from Monday to Saturday. It's right by Gants Hill tube station.

The Byte Shop 426/428 Cranbrook Rd., Gants Hill, Ilford, Essex. Telex 897311
Telephone 01-554 2177

Letters

PUZZLE DAZZLE

Here is a little puzzle to test your readers' 8080 machine code ingenuity:
"Imagine you possess an 8080 with 64K bytes of read/write memory which you want to clear. Write a program that sets all 65536 bytes to zero."

You may be able to afford a nominal prize. If so, in the event of more than one correct answer the prize goes to the person who gives the solution of minimum size; failing that, of minimum execution time. If no one solves it, the prize goes to me! (Assumin, of course, that I can produce an answer, which I believe I can.)

David C. Broughton,
38 Westbury Road,
Northwood, Middlesex HA6 3BX

PCW All right, readers. Prize is £2. **PCW**

GO AHEAD – BLAME US FOR EVERYTHING!

Winning a 'Sinclair Programmable' (**PCW** Competition in Issue 1) presented me with a bit of a problem. What do I do with two Sinclair Programmables? I bought one shortly after they were first available. My solution was to interface one of them to my MPU system. I now have a powerful but relatively slow extension to my micro-computer equivalent to a floating point package complete with scientific functions. Since PCW caused the problem I would be willing to send you details of the interface if you think it may be of interest to others.

W. McIvor,
2 George Street,
Accrington, Lancs BB5 0HD
PCW Let's have a look at your article. Soon. **PCW**

'PRECISELY SPEAKING',
Sheridan Williams, (PCW September 1978)

In reply to the above article:—

Computers **do** make mistakes! (sometimes).

Even the hardware self-correction features of the most modern computers do not detect all hardware errors. Some errors get past these traps and the machine may for example lose bits from a register, which could lead to erroneous results. Also it is not unknown for design errors to be found after a machine has been widely sold, e.g. the recall of the HP-35 calculators due to pattern sensitivity in semi-conductor memory.

The exhaustive testing of complex hardware and software is such an immense undertaking that it is rarely if ever done! So do not believe that the computer is infallible, it is the product of mere humans who are notoriously liable to error.

C.G. Hurnell,
10 Farm Close,
Crowthorne, Berks.

Adding Machines?

I am a computer user both at work and at University. As a user, I have found your magazine most interesting and am now considering purchasing a personal computer. The main reason being that developing programs for playing chess on a circular board (well I believe it's possible) and other such games is somewhat frowned upon, particularly at work.

The languages I use mostly are Fortran and Basic. After using these languages, it strikes me that some of the personal computers in your magazine are not much more than adding machines. I mean no offence by this, however a machine that can handle only one array and 26 numerical variables is a little limiting. For a few hundred pounds, though, the machine can be classed as good value as a powerful calculator.

Not being particularly rich at the present time, I was most interested in Sheridan Williams' letter mentioning Elliot computers.

I would be most interested to hear from anybody who has details of these machines, in the way of practical handling capabilities and I would also be interested in purchasing possibilities.

Articles in your publication, which tell the computer freak more about fundamental specifications of computers, may be helpful to many readers like myself who are interested but not, at present, very knowledgeable when it comes down to 'how good is a computer?'.

M.D. Newton,
127 Chatsworth Road,
Morecambe, Lancs.

I have been following with interest the reviews in PCW, acquiring manufacturers hand-outs and even had a "hands-on" a PET for a short time.

I am thus coming to the conclusion, which I hope can be refuted, that what I call "Graphics" on a main frame or mini-computer i.e. plotting graphs and histograms, is not the same as the Graphics on most of the micros currently offered (which is a primitive form of art work).

I write as an engineer who merely regards a computer as a tool for generating information in the most digestible form. This is universally accepted as graphs or histograms. The PET with the hardware potential of a useful engineering/scientific computer is only capable of playing games and acting as a programmable calculator. The APPLE has most exotic graphics and no transcendental functions!

The manufacturers would be surprised at the number of organisations with main frame machines which are only used for pay roll and stock control, and which engineers are not allowed to use. I believe many engineers would personally purchase a machine of the capability indicated.

The manufacturers of micros seem not to have realised that they could displace the scientific calculator as decisively as the calculator displaced the slide rule if they would only provide the software needed. It should be remembered that most engineers are not too good at writing software at the low level of a plotting package – besides, they don't have the time.

P.S. I know about the Tektronix 4051 and the Hewlett-Packard 45 and I can't afford either.

A.B. Bailey, D.Phil (Oxon),
Endrickvale,
Fintry, by Glasgow

Dr. Johnson's reply

I was sorry to see that Dr. Price (PCW August 78, page 8) has lost the euphoria of the halcyon days of the 1960's: I for one, most assuredly have not. My criticism of the DHSS is precisely because they failed to apply the criterion so aptly suggested by the Journal Dr Price quotes. The Medol system in which I have recorded every sign, symptom, diagnosis and treatment since 8 August 1970 requires distinctly less than the "12 key words" specified: each general practice consultation occupies only 40 bytes, on average.

Dr Michael Hendry (PCW October 78, page 10) may be similarly surprised at the degree of data compression that becomes available using Medol; though he may be interested to hear that analysis of such data shows that "cough" is the commonest symptom (21% of all symptoms) apart from pain (25%). Confidentiality, which is of supreme importance, ceases to be a problem with decentralised systems: you simply lock away the cassettes.

Using currently available hardware, a system for computerising GP's paperwork will be available from DOC, P O Box 99, London SW16 2DL, at the end of October 1978. It will be a long time before other medical systems can compare with the detailed clinical analysis available with Medol, but not long before technology becomes cheap enough to equip each GP's desk with a 16-bit micro, using it.

Robert Johnson, MSc (computing) MRCGP.
"Thorns"
16c Clough Lane,
Grasscroft, Oldham, Lancs. OI4 4EW.

PCW Dr. Johnson's article (in Vol 1, No 2), the subsequent correspondence and the different approaches being taken towards the use of Computers in medicine, are healthy signs of the urgency with which informed people are treating a problem which has so far seemed intractable. In our next issue we present another approach, in it Nigel Bevan of the National Physical Laboratory writes on MICKIE,. **PCW**

THE RESEARCH MACHINES 380Z COMPUTER SYSTEM



THE RESEARCH MACHINES 380Z
A UNIQUE TOOL FOR RESEARCH AND EDUCATION

Microcomputers are extremely good value. The outright purchase price of a 380Z installation with dual mini floppy disk drives, digital I/O and a real-time clock, is about the same as the annual maintenance cost of a typical laboratory minicomputer. It is worth thinking about!

The **RESEARCH MACHINES 380Z** is an excellent microcomputer for on-line data logging and control. In university departments in general, it is also a very attractive alternative to a central mainframe. Having your own 380Z means an end to fighting the central operating system, immediate feedback of program bugs, no more queuing and a virtually unlimited computing budget. You can program in interactive BASIC or, using our unique Text Editor, run very large programs with a 380Z FORTRAN Compiler. If you already have a minicomputer, you can use your 380Z with a floppy disk system for data capture.

What about Schools and Colleges? You can purchase a 380Z for your Computer Science or Computer Studies department at about the same cost as a terminal. A 380Z has a performance equal to many microcomputers and is ideal for teaching BASIC and Cesil. For A Level machine language instruction, the 380Z has the best software front panel of any computer. This enables a teacher to single-step through programs and observe the effects on registers and memory, using a single keystroke.

WHAT OTHER FEATURES SET THE 380Z APART?

The 380Z with its professional keyboard is a robust, hardwearing piece of equipment that will endure continual handling for years. It has an integral VDU interface – you only have to plug a black and white television into the system in order to provide a display unit – you do not need to buy a separate terminal. The integral VDU interface gives you upper and *lower* case characters and low resolution graphics. Text and graphics can be mixed *anywhere* on the screen. The 380Z has an integral cassette interface, software and hardware, which uses *named* cassette files for both program and data storage. This means that it is easy to store more than one program per cassette.

Owners of a 380Z microcomputer can upgrade their system to include floppy (standard or mini) disk storage and take full advantage of a unique occurence in the history of computing – the CP/M™* industry standard disk operating system. The 380Z uses an 8080 family microprocessor – the Z80 – and this has enabled us to use CP/M. This means that the 380Z user has access to a growing body of CP/M based software, supplied from many independent sources.

380Z mini floppy disk systems are available with the drives mounted in the computer case itself, presenting a compact and tidy installation. The FDS-2 standard floppy disk system uses double-sided disk drives, providing 1 Megabyte of on-line storage.

**Trademark, Digital Research.*

Versions of BASIC are available with the 380Z which automatically provide controlled cassette data files, allow programs to be loaded from paper tape, mark sense card readers or from a mainframe. A disk BASIC is also available with serial and random access to disk files. Most BASICs are available in erasable ROM which will allow for periodic updating.

If you already have a teletype, the 380Z can use this for hard copy or for paper tape input. Alternatively, you can purchase a low cost 380Z compatible printer for under £300, or choose from a range of higher performance printers.

380Z/32K complete with SINGLE MINI FLOPPY DISK SYSTEM MDS-1
£1787.00

380Z/16K System with Keyboard
£965.00

RESEARCH MACHINES Computer Systems are distributed through SINTEL, P.O. Box 75, Chapel Street, Oxford. Telephone: OXFORD (0865) 49791. Please contact SINTEL for the 380Z Information Leaflet. Prices do not include VAT @ 8% or Carriage.

Tid Bits PRODUCTS . . . . COMPANY NEWS . .

THE CONTINENT: NEW COMPUTER AND COMPUTING SHOW

Industrie und Personal Computer Expo of Munich, West Germany is to hold its first Industrie and Personal Computing Show in Munich from the 9th to 15th of Nov. 1978.

The Show will coincide with ELECTRONICA, held a few hundred yards away.

On Show will be industrial products as well as European and American books and magazines.

For further information:
KILOBAUD – Europe uP-Literatur
Marktstrasse 3, D – 7778 Markdorf *or*
ABC – Computershop Schellingstrasse 33
D – 8800 Munich 40.

PET GAMES

MINI – MICRO COMPUTER GAMES of 47, Queens Rd., London, N11 2QP (01-889 7615) has published a new catalogue of games available on High Quality C12 Cassette Tapes on CBM PET Standard. Fully documented.

Further information and a copy of the catalogue can be obtained from the address above.

Low-cost Paper Tape Reader

Microsystem Services announce that they can supply a stand-alone paper tape reader.

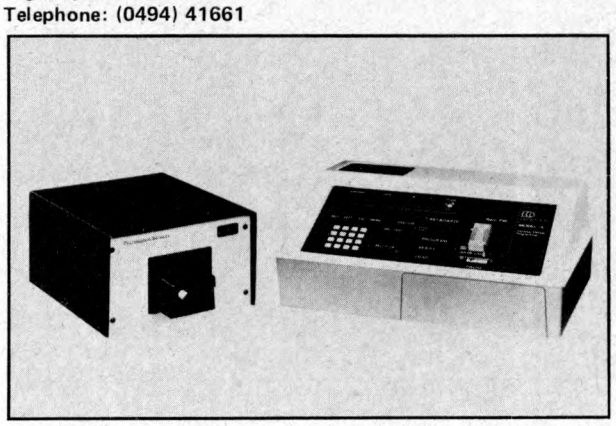
Known as the MSS96R, the reader can handle both the 8-level codes commonly used in data processing and the 5-level (Baudot) codes often employed in data communications.

Both parallel and serial interfaces are included. In the serial mode the data word format is switch-selected.

The baud rate can be switch-selected to any of seventeen values between 50 and 9, 600 baud.

Microsystem Services have the MSS96R tape reader in stock and can arrange immediate delivery. The one-off price is £680.

For further information:
Jim Knott
Microsystem Services
Duke Street
High Wycombe, Bucks
Telephone: (0494) 41661



The MSS96R stand-alone paper tape reader from Microsystem Services.

NEW DOUBLE SIDED, DOUBLE DENSITY FLOPPY DISC DRIVE FROM BASF

The 6104 Floppy Disc Drive allows single or double density recording on one or two sides of a standard 8" diameter floppy disc. The maximum capacity of a disc is 12.8 Million bits, unformatted, using an MFM or M^2 FM recording method.

Head positioning time is only 3 milliseconds, and the unit measures:

110(H) x 220(W) x 360(D) mm

For further information:
M.J. Robinson
Datac Limited, Tudor Road, Altringham
Cheshire WA14 5TN

NEW CMOS RAM FROM RCA

The MWS5101 Series 256-word x 4-bit static random-access memories are now available from RCA Solid State with a choice of four different options on speed and leakage current. Using RCA's silicon-on-sapphire C-MOS integrated-circuit technology, which combines high-speed operation with the low-power, high-noise-immunity advantages of C-MOS circuitry, the memories offer access times of 250-450ns and quiescent currents from only 10uA to 500uA. All the devices are fully characterised for an input voltage of 5V over the temperature range 0°C to $+70^{o}$C.

Designed for use in memory and microprocessor systems where high speed, very low operating current (4mA at 5V) and simplicity of use are desirable, the MWS5101 has separate data inputs and outputs and operates from a single power supply of 4.75 – 5.25V.

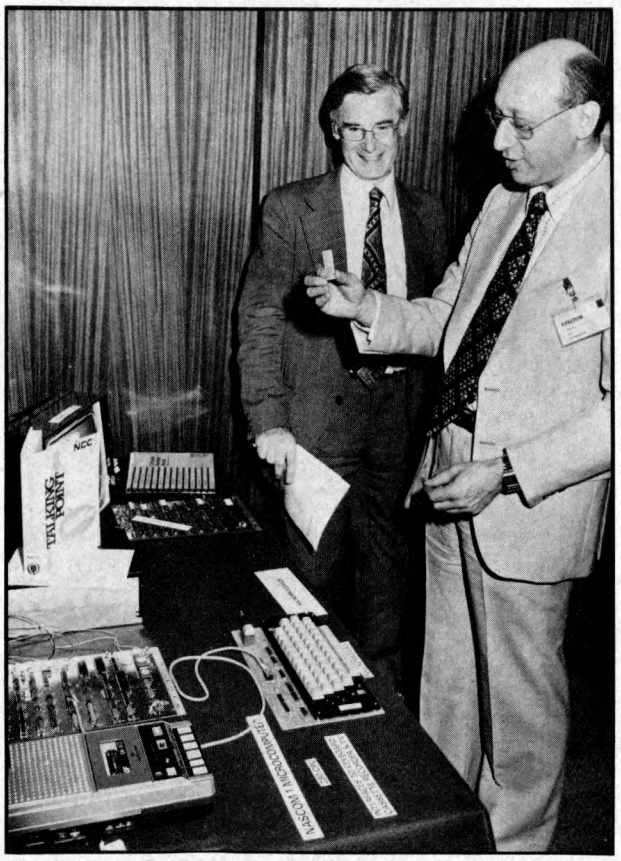
An important feature of the MWS5101 is the low-voltage data-retention capability of 2V. The memory is compatible with TTL circuitry, and for TTL interfacing at 5V operation excellent system noise margin is preserved by using an external pull-up resistor at each input.

More information: RCA Solid State – Europe,
Sunbury-on-Thames,
Middlesex, TW16 7HW.

Industry Minister Visits NCC

The Rt. Hon Gerald Kaufman MP, Minister of State, Department of Industry, visited The National Computing Centre on 28 July.

During his visit Mr Kaufman, whose constituency is Ardwick in Manchester, discussed the Centre's work programme, particularly that related to Microprocessors and Training. He saw a small display of equipment that included the NASCOM 1 – a small British supplied microcomputer kit that can be linked to a domestic TV monitor and a cassette recorder.



The Rt. Hon Gerald Kaufman, MP, holding the microprocessor taken from the board of the Nascom 1 Microcomputer during his visit to The National Computing Centre on 28 July 1978. With him is David Firnberg, Director of the NCC.
NCC, Oxford Road, Manchester, M1 7ED 061-228 6333

BREADBOARD '78

Breadboard – a show for the home enthusiast – is to be held in Seymour Hall, Seymour Place, London, W1 between Nov., 21st and 25th, 1978.

The program includes demonstrations, games and competitions.

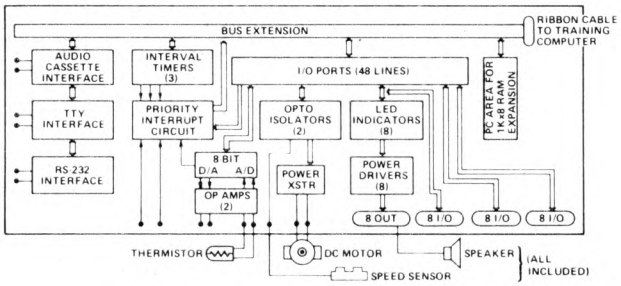
Further information from:
Trident Conferences & Exhibitions Ltd.,
Abbey Mead House,
23a Plymouth Road,
Tavistock,
Devon PL19 8AU
Tel: 0822 4671 Telex: 45412 TRITAV

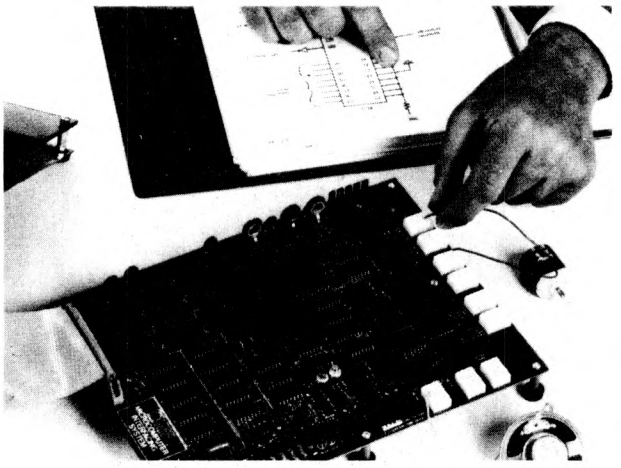
SELF-STUDY COURSES

Integrated Computer Systems offers Microcomputer Self-study Training Courses. It provides "hands-on" software/hardware training.

For further information:
ICSP Inc – UK Computer Marketing, 15 Bevis Marks,
London, EC3 7LH Phone: (01) 283 3974

MICROCOMPUTER INTERFACING SYSTEM





EXIDY'S NEW PERSONAL COMPUTER

Exidy Incorporated, the third largest producer of video arcade games, has entered the consumer electronics market with its new user-programmable personal computer, the Sorcerer TM.

The self-contained Sorcerer needs only to be plugged into a video display and a cassette tape recorder to be a fully-functioning computer system.

Exidy introduces a new concept in user-programmable personal computers with the Sorcerer's exclusive plug-in Rom Pac TM cartridges. The unique Rom Pacs contain high-level programming languages, operating systems or special proprietary software.

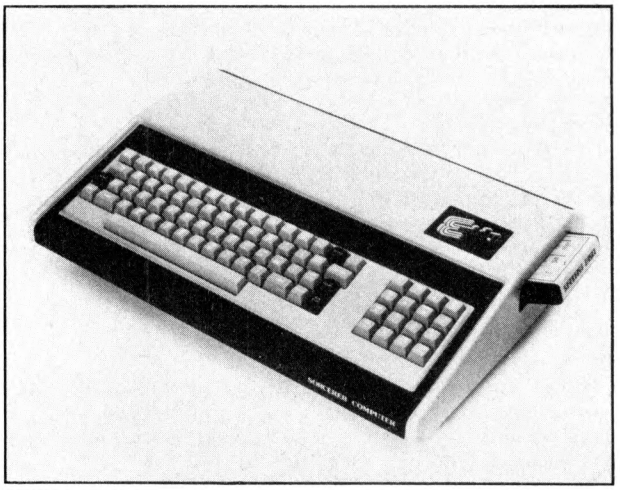
Each Sorcerer comes with a Rom Pac cartridge containing Standard BASIC. Additional Rom Pacs available or now in development include a user-programmable EPROM Rom Pac, an assembler editor, a disc operating system and a word processing package.

The user can change the Sorcerer's operational function by either plugging in a different Rom Pac, or, with standard BASIC and other programming languages, he can create his own programs and modify existing ones.

The Sorcerer offers both serial and parallel interfaces and modem control, and is based on a Z-80 uP.

Exidy will market the Sorcerer through a retail network of computer stores, consumer electronics merchandisers and electronics shops both domestically and overseas. European video and power requirements can be factory set.

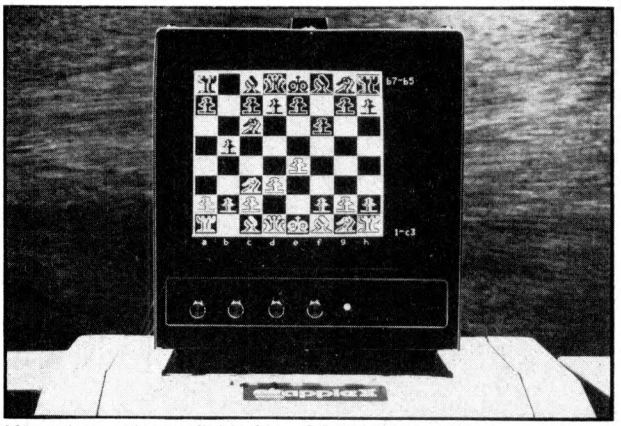
For further information:
Comp Computer Components
Freepost, Barnet, Herts.
Tel 01-441 2922



The Sorcerer

CHESS FOR PET, TRS-80, APPLE

MICROCHESS 2.0 for 8K PETs and 16K APPLEs, in 6502 machine language, offers 8 levels of play to suit everyone from the beginner learning chess to the serious player. It examines positions as many as 6 moves ahead, and includes a chess clock for tournament play. MICROCHESS 1.5 for 4K TRS-80s, in Z-80 machine language, offers 3 levels of play. Both Level 1 and Level II versions are included and can be loaded on any TRS-80 without TBUG. MICROCHESS checks every move for legality and displays the current position on a graphic chessboard. You can play White or Black, set up and play from special board positions, or even watch the computer play against itself! MICROCHESS is prices at $19.95 and is available from authorized dealers worldwide or by mail from Personal Software, P.O. Box 136, Cambridge, Mass. 02138 U.S.A., tel. (617) 783-0694. VISA/Bankamericard and Master Charge orders accepted.



Microchess: Also available from PETSOFT in Britain.

NEW MAGAZINE

A new weekly news magazine serving the rapidly-expanding computer industry is to be launched on September 27 by a new publishing company.

The new publication – *Computer Talk* – will face stiff competition from three existing weeklies, Haymarket's *Computing* and *Datalink* and IPC'S *Computer Weekly*.

Computer Talk is the brain-child of Wendy Hutton (27) and Jackie Wild (25) who both left the computing papers at Haymarket in order to set up the new company, Talk Publications Ltd. A staff of 20, many of them coming from Haymarket, has been engaged to work on the launch of the new magazine.

Editor of *Computer Talk* will be John Sterlicchi, formerly assistant editor of Haymarket's *Accountancy Age*. He was at one time, news editor of *Computing*.

780K BYTES ON-LINE STORAGE IMSAI'S VDP-4X INTEGRATED SYSTEMS

IMSAI MANUFACTURING CORPORATION announces **SERIES VDP-4X** Video Data Processor. Three versions, the VDP-40, VDP-42 and VDP-44, provide a disk storage capacity of **180K, 400K and 780K bytes,** respectively.

A fully integrated system, the VDP-4X features an 8085 microprocessor, 32K/64K RAM, dual 5¼-inch floppy disks, 9-inch CRT, keyboard and serial I/O in a desk-top cabinet.

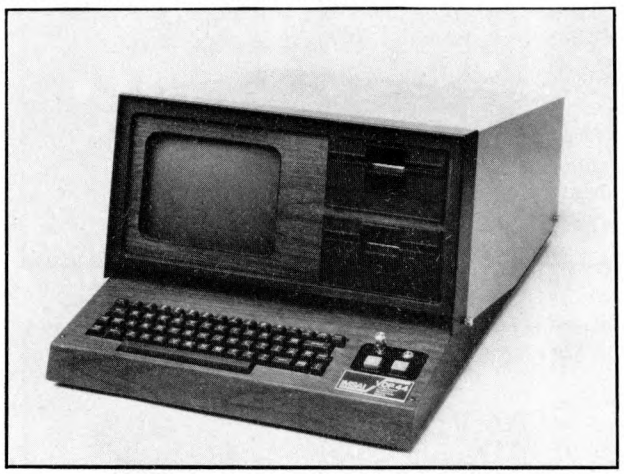
IMSAI's multi-disk operating system (IMDOS) is provided with the VDP-4X and has a large number of utilities – including an 8080/85 assembler, video/context editor, dynamic software debugging program and floppy disk system diagnostic program.

780K bytes on mini-floppies is achieved by combining the IMDOS disk formates with double density, 77-track mini-disk drives.

The VDP-4X can communicate with other VDP-4Xs and other computers, or act as a terminal in a data communication network. Virtually unlimited applications are possible. For example, use the VDP-40 for distributed data processing applications. Or computerize the accounting and inventory functions of a small business with the double density VDP-42. Select the double density, double track VDP-44 where greater disk storage capacity is required.

Serial I/O is used to drive optional peripheral devices, including line printers, modems and auxiliary terminals.

IMSAI MANUFACTURING CORPORATION
14860 Wicks Blvd., San Leandro, CA 94577
(415) 483-2093 TWX: 910-366-7287



Micro Focus Announce CP/M Version of CIS COBOL for Personal and Small Business Computers

Individuals and small businesses wishing to purchase the new version of CIS COBOL will require an 8080 or Z-80 based computer with 32K bytes of RAM and a CRT terminal and should be using the CP/M operating system. CP/M has become established in the USA as the de facto standard operating system for S-100 system and is rapidly gaining popularity in the UK.

COBOL is a standard high-level language in use throughout the world for business applications. CIS COBOL, developed by the British company Micro Focus, is a sub-set of ANSI COBOL 74 and is probably the only COBOL compiler that will run on a system with as little as 32K bytes of memory. CIS COBOL is supplied with a library of CRT driver programs to make it easy for the user to configure it to suit his own particular CRT without having to write interface software. CIS COBOL contains optional extensions to provide an interactive capability and also features interactive debug. This interaction makes it easy to develop software quickly.

For further information:
Paul O'Grady or Brian Reynolds
Micro Focus Limited
18 Vernon Yard
Portobello Road
London W11 2DX
Telephone: 01-727 5814

COPYRIGHT

Please note that the name TELETYPE is copyrighted by the Teletype Corporation.

Also copyrighted is the name EURO-BUS. It is the trade name for part of the product range of Mektron Circuit Systems Ltd., of 271 Kingston Rd., Leatherhead, Surrey.

THE MICROCOMPUTER INVESTORS ASSOCIATION

It is a non-profit organization whose purpose is to facilitate the exchange of data and information relating to investments and microcomputers with the express purpose of such interchange being directed toward maximizing profits in stocks, bonds, warrants, stock options, and commodities.

In order to become and remain an active member of the Microcomputer Investors Association each associate must pay dues (currently $30 per year) and submit at least one article per year suitable for publishing in the Association's journal, *The Microcomputer Investor.*

To obtain a Membership application, send a *Self-Addressed Stamped-Envelope* to:

Microcomputer Investors Association, 2415 Ansdel Court, Reston, Virginia 22091

PET Software

From our latest catalogue of over 70 titles:

General Ledger	£25
Program Linker	£10
Stock Control	£12
Word Processing	£12
Cassette Filing Tutorial	£ 8
Assembler Editor	£25
Line Renumber	£20
Super Startrek	£ 8
Backgammon	£ 8

Plus over 65 more in our latest catalogue. If you are a PET user, send us an SAE and we will add your name to our mailing list. Credit Card orders accepted by telephone.

Petsoft

PO Box 9, Newbury, Berks. RG13 1PB
Tel. 0635-201131 01-352 1100 Telex 8951672

EQUINOX 300

A powerful multi-user
multi-tasking
multi-language

16-bit microcomputer time-sharing system

supporting
* BASIC
* LISP
* PASCAL
* Floppy Discs
* Hard discs

including a powerful Text Formatter, Assembly Language Development System and disc based Sort utilities.

Priced from under £5,000

Write or phone for further information

EQUINOX COMPUTER SYSTEMS LTD.
32-35 Featherstone Street,
London, EC1Y 8QX
(01-253 3781/9837)

Pet Sounds

Just connect an Amplifier or Cassette Recorder to the back of your Pet and add sound effects and tunes to your programmes. No Interface necessary.

The following programmes are available on cassette:—

1. MOON LANDING £6 (With Sound Effects)
2. TUNES I £3 (Pop + Folk)
3. TUNES II £3 (Traditional)

Cheque or P.O. to:— J. H. MacLeod, 11 St. Martins Drive, Eynsford, Kent, DA4 0EY.

A COMPUTER FOR SMALL BUSINESS AT £695

See and try the Commodore PET 2001 Computer that has broken the price barrier and swept the USA (as featured in "Personal Computer World"). Designed for ordinary people with no previous experience. For payroll accounting, stock & inventory control, personal & business records. Completely self-contained with video display and keyboard for desk use. Full service guarantee and advice backing. Free seven lesson cassette course with every PET. HP terms available.
Demonstrations daily: phone Catherine (01-930 1638) or call Mons-Sats. SAE brochure.

TLC WORLD TRADING LTD
32 Craven Street, Charing Cross, London WC2

ENTHUSIASTS OF PET

This is what you have been waiting for.
PME 1 24K Memory Expansion Board for Commodore Pet Computer. From Justwise Computer Systems. At retail price of £380 Plus V.A.T. (Discounts for orders over 10)

The PME 1 is designed specifically to fit into the Pet Computer. All hardware required is furnished with the board. Absolutely no electrical modifications to the Pet are required. The Memory Derives its power from Pet Transformer and not from its regulators. The 24K Board allows you to write programmes to the total capacity of the Pet.

The PME 1 Memory Expansion Board is warranted against defects in materials and workmanship for a period of six months from date of purchase.

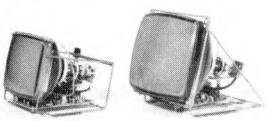
First come first served. WISE

Contact Ray Naik at:—
Justwise Computers Systems Limited.,
1-11 Hay Hill, London W.1.
Tel: 01-493 7875.

TV MONITOR KITS

VISIONKITS

gives you professional TV equipment in kit form for all data display and picture applications, providing far superior results to converted TV receivers.



Top quality monitors exactly as produced by one of today's leading manufacturers. All parts plus comprehensive assembly instructions supplied, along with full application notes, 9" or 12" tube plus chassis kits are available with a separate printed circuit module kit to fit other size of tube.

SPECIFICATION:
Video — Composite 1V ± 6dB Input impedance 10K ohms
Power — + 12 volts stabilised Current 1 amp
System — 625 lines 50 fields and 525 lines 60 fields
Video response to 10MHz
Operating temperature — 0 — 50°C

PRICE: (Including VAT and inland postage and packing)

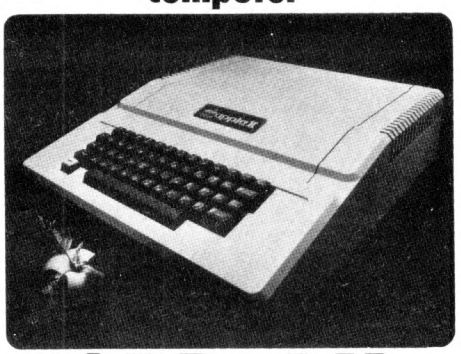
9" tube and chassis kit	£45.00
12" tube and chassis kit	£45.00
P.C. module kit	£55.00
Ready-built 9" monitor	£150.00
Ready-built 12" monitor	£150.00
Mains power supply for 12" monitor (built)	£40.00

Please allow 21 days for delivery
Send s.a.e. for copy of application notes only
Mail order only.

VISIONKITS

9 Claymill Road, Leicester LE4 7JJ

The world's best-selling personal computer



APPLE II

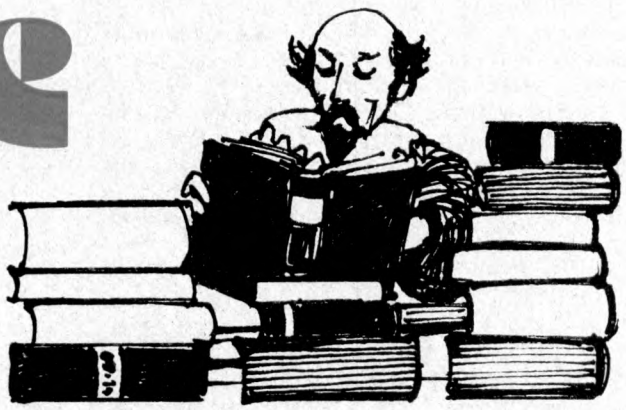
***New powerful basic and new graphics on Rom. £75 card.**
***Talk to Apple with voice recognition, speaker trained, 32 word vocabulary. £165 card.**
***Colour Graphics. High resolution 280h x 192v, 6 colours, easy-to-use. Low resolution 40h x 48v, 16 colours, very powerful.**
***Apple's disks. Powerful DOS. 116K bytes capacity, multiple drives, fast access. £395.**
***Use Apple as a computer terminal 110 or 300 BAUD. Full or half duplex or use with a Dec-writer. £95 card.**
***Use any 8 bit parallel printer with Apple 11. Print up to 3,700 lines per minute. 255 character lines, upper and lower case. £100 card.**

Personal Computers Limited

DISTRIBUTOR
18-19 Fish Street Hill, London E.C.3.
Tel. 01-283 3391
New address from August 21st
194-200 Bishopsgate, London E.C.2.

THE BIG APPLE 48K RAM·18K ROM £1850 incl.VAT

The Micro Muse



"Shall I compare thee to a digital display..."

Eric Finlay

Eric Finlay studied at the Slade School in London, and the Hochschule fur Bildende Kunst in Berlin. His graphic work has been shown at the Museums of Modern Art in Paris, Lubljana, and Tokyo. He has taught printmaking at Chelsea School of Art since 1959. His interest in computers dates from around 1968 when he designed, built and demonstrated a simple L.S.I. graphics computer at Chelsea.

Computer poetry is a development of literacy specific to the second half of the present century. The evolution of the microprocessor has put a powerful new technology into the hands and minds of thoughtful and creative people. It is apparent that the conditions now exist for a renaissance of creative writing along these lines. The difference between contemporary computing and that of a decade ago is that experimental work is no longer conditional upon access to large computer installations.

Joseph Weizenbaum, in an amusing and pathetic chapter of his book 'Computer Power and Human Reason' (hailed by Lewis Mumford, in Scientific American, as "A landmark in the integration of scientific and humanistic thinking"), says:—

"**Wherever computer centres** have become established, that is to say, in countless places in the United States, as well as in virtually all other industrial regions of the world, bright young men of dishevelled appearance, often with sunken glowing eyes, can be seen sitting at computer consoles, their arms tensed and waiting to fire their fingers, already poised to strike, at the buttons and keys on which their attention seems to be riveted as a gambler's on the rolling dice. When not so transfixed, they often sit at tables strewn with computer printouts over which they pore like possessed students of a cabalistic text. They work until they nearly drop, twenty, thirty hours at a time. Their food, if they arrange it, is brought to them: Coffee, Cokes, sandwiches. If possible, they sleep on cots near the computer. But only for a few hours – then back to the computer console or the printouts. Their rumpled clothes, their unwashed and unshaven faces, and their uncombed hair, all testify that they are oblivious to their bodies and to the world in which they move. They exist, at least when so engaged, only through and for the computers. These are computer-bums, compulsive programmers. They are an international phenomenon."

On the cover of the same book, Mr Weizenbaum states with diagnostic gloom:—

> "Science promised men power. But, as so often happens when people are seduced by promises of power, the price actually paid is servitude and impotence"

Whatever one may conclude about the conditions prevailing at M.I.T., it seems evident that computing, as an archetypal function of the mind, can release and canalize creative energy; and also induce what might be termed 'transcendental' states of mind.

This must be of considerable interest to poets.

Computer poetry has a few of its roots in the experimental electronic work of the Italian Futurists, but the direction may be traced as far back as 'The Gold Bug' by Edgar Allen Poe, in which the theme is the decoding of a message found by the author in the jaws of a beetle. Such a story quite naturally emerged as a consequence of the work done by Samuel Morse in preparing a language suitable for long distance communication by elec-

tronic means. In 'The Thousand-And-Second Tale of Scheherezade' Poe gives an amusing, if sinister, picture of 19th Century technology as seen through the eyes of a poet.

The interested reader is referred to an excellent essay on the science fiction of Poe by Harold Beaver (The Science Fiction of Edgar Allen Poe. Penguin. 1976) for an insight into the creative mood prevalent in Poe's time. Baudelaire's translations of the works of Poe had an immense influence on his own writing, and his experiments with 'poetic-prose' are a direct result of translating works like 'A Descent into the Maelstrom', and 'A Man of the Crowd'. These experiments influenced Mallarme, Rimbaud, Huysmans, Lautreamont, and continued into the present century with the work of Apollinaire, Arp, Schwitters, Breton, Joyce, Pound, Beckett and Borges to name a distinguished (and almost extinguished) few.

The fusion of mathematics and art has a much more ancient history, and can be traced at least as far back as the Pyramids. Poets have often chosen to work in metrical confines (Virgil, Dante, Shakespeare, Milton, Pope, for example), and perhaps the most extreme example of metrical limitation, used creatively, is the Haiku of Japan, which influenced the Imagist School in England to move in the direction of simplicity, and compression of meaning.

Jacia Reichardt, in her book 'The Computer in Art' *(Studio Vista, 1971)* claims computer poetry to be an offshoot of 'Concrete Poetry'. This is not very precise, and I prefer to regard computer poetry as verse composed for performance by a computer (sometimes even composed *by* a computer), which is not characteristic of other types of poetry.

The computer poem can be appreciated on several levels. It can exist as a pleasing, or, at least puzzling arrangement of symbols on paper, as mysterious and challenging as a tablet of Assyrian script; it can be translated into the designed image without a computer by an intelligent reader (who is thus creatively involved in the poem in an intellectually satisfying way); or it can be translated directly into its kinetic equivalent by performing it on the computer.

To illustrate these possibilities I have composed probably the smallest kinetic poem ever written. It occupies one segment of an 8-bit L.E.D. display. I apologise to the Japanese for thus making the Haiku seem long-winded.

MICRO-POEM 1

```
                    ; 7-segment 8-bit LED image
                    ; SC/MP 11

OF20    C2          @disp(ptr)
OF21    00          disp
OF22    C9          @disp(ptr)
OF23    04          disp
OF24    90          JMP
OF25    FA          disp
.....   .....
OFF9    0D          P1 (H)
OFFA    00          P1 (L)
OFFB    OF          P2 (H)
OFFC    40          P2 (L)
.....   ....
OF40    D3          image
oooo    oo
                    ; end
                    ; e.f.
```

★★★★★★★★★★★★★★★★

Anyone with an SC/MP 11 Micro can program the above text into the computer and it will substantiate the modest claim I have made for it. For those who do not yet possess a computer, all is not lost. All that is required is a pencil and paper and a knowledge of the code by which alphanumeric data can be put into a computer, for translation into 7-segment display form. I will discuss the display code here, since it may be of interest to those who are still grappling with the problem.

The key information is contained in the program at the address OF40, and is D3, which is hexadecimal code for the decimal numbers 13 and 3.

i.e.

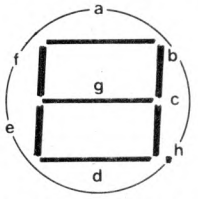
0	1	2	3	4	5	6	7	8	9	10	11	12	**13**	14	15	Decimal
0	1	2	3	4	5	6	7	8	9	a	b	c	**D**	e	f	Hexadecimal

The computer can only read binary code, and so the true nature of the signal from the computer's point of view, is:—

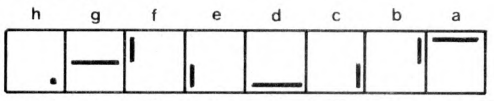
8	4	2	1	/	8	4	2	1	Powers of 2.
1	1	0	1	/	0	0	1	1	**Binary equivalent**
		13		/			3		Decimal equivalent.

The decimal equivalent is found by adding together powers of 2, signified by 1, or 0, so that 0011 means 2 plus 1, equals 3 (Hex) or decimal; and 1101 equals 8 plus 4 plus 1, equals 13 decimal, or D, in hexadecimal.

A 7-segment display element, common to most calculators, and some micro-processor systems usually contains 8 diode elements (one for the decimal point).



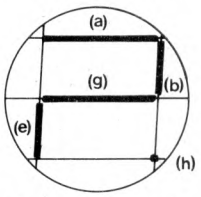
When the computer 'sees' the signal 11010011 , it is instructed by the *rest* of the programme to light up the diode elements in the display unit according to the following code:—



1	1	0	1	0	0	1	1	Binary
			13				3	Decimal
			D				3	Hexadecimal

(1 = lit up, 0 = not lit up)

If the binary elements of the hexadecimal signal D3 are added together in the **same unit,** the results will be:—



Within the small area of an LED display unit, the above image gives an acceptable equivalent to the question-mark.*

The reader should now be able to translate any machine-code data into plain English (e.g. try 66, 06, 73, 73, 79, 79, 82, 82.). If he prefers to encode the Chinese alphabet for computer, or experiment with Runic Script, the way is now clear.

In my next article I will show how the rest of the instructions in the micro-poem programme persuade the computer to activate the display. I will also illustrate my second micro-poem, which occupies the full 8-bit display, and show how this can be programmed.

* *Following the example of James Joyce, I leave the interpretation of this poem to future generations of literary critics.*

PCW MICROCOMPUTER CHESS CHAMPIONSHIP

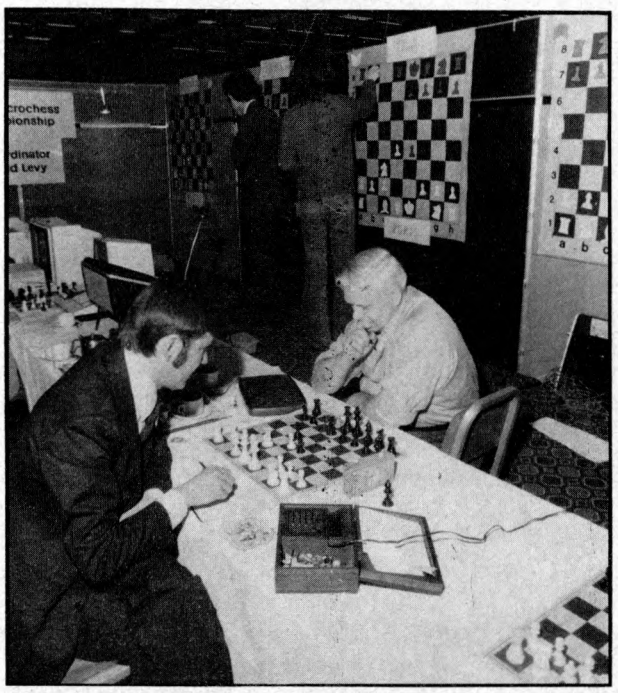
David Levy

As part of the PCW show at the West Centre Hotel in London, it was decided to organize the first microprocessor chess tournament to be held in Europe. Only one other event of its kind had ever taken place; that was in California earlier this year. Now that many individuals are writing chess programs for their own computers we shall doubtless see many such events in the future. Computer tournaments provide an excellent medium for chess programmers to exchange ideas and to learn from each others efforts. Also, they are great fun!

The tournament in London was an all-play-all event with six contestants. Three of the entries were from England, all written by private individuals who had done the programming in their own time. The other entries were from North America and were all the result of commercial enterprise – Boris and Chess Challenger are marketed as stand alone units while Microchess is sold by a personal software company based in Toronto, and can run on various microprocessors (Tandy, Commodore). Some interesting details about each program can be seen in figure 1 below.

Program	Programmer	Machine	Positions examined per sec.	Language
MIKE (England)	Mike Johnson	Home built Motorola 6800	300	6800 assembler
BORIS (U.S.A.)	?	F8	approx 30	F8 code
CHESS CHALLENGER (U.S.A.)	?	Z80A	35	?
MICROCHESS 2.0 (Canada)	Peter Jennings	Commodore Pet	?	6502 machine code
FAFNER (England)	G.A. Burkill	Apple II 6502	2,000	Assembler/ Basic
COCMA (England)	A. Cornish	Apple II 6502	20	Assembler/ Basic

I was very pleased when PCW editor Meyer Solomon invited me to act as tournament director and commentator. I had filled this post in many earlier computer chess tournaments though never in an event in which all the entrants were running on micros. Before the tournament began I was not particularly optimistic about the standard of play that could be expected from these programs but I must confess that I was pleasantly surprised when I discovered that the best programs were playing at about the level of the average mainframe programs of a decade ago. Presumably, since micros are readily available to almost anyone with a yen for programming, many more chess programs will be written for home computers during the next few years. It would therefore seem reasonable to expect that the standard of play will increase substantially and that within a decade there will be matchbox sized machines that can play



Boris plays Chess Challenger. Both machines were in at the kill.



A tense moment, MIKE v. FAFNER (extreme left of photo).

chess as well as the current World Computer Champion, CHESS 4.7.

The first round of the tournament produced two unfortunate incidents. In the game between FAFNER and MIKE, MIKE was forced to abandon the game in a winning position because the voltage was slipping down to 220 and the machine on which MIKE was running required 240 volts. Mike Johnson rushed across London during the lunch interval to fetch a transformer which would step up the voltage and by the afternoon he was back in action. After this misfortune MIKE played excellent microchess for the remainder of the event.

The other *domage* occurred in the game between BORIS and CHESS CHALLENGER, the world's two best known commercially available chess machines. During the game CHESS CHALLENGER played the perfectly reasonable move Pb7-b6 (pawn on b7 moves to b6), which was the most logical move in the position and the move which I had predicted in my commentary to the audience. Later in the game White (BORIS) played its queen to c4, giving check, whereupon CHESS CHALLENGER tried to capture the queen with something on b5. When the position was verified it transpired that C.C. thought that it had a pawn on b5 instead of b6. Since, at that point in the proceedings, C.C. had only one minute remaining before the time control, it was quite impossible for its operator to recover the situation.



David Levy and the PET chess program developed by Personal Software (now available from Petsoft).

After the first round BORIS, MIKE and CHESS CHALLENGER clearly demonstrated their superiority over the other programs. When the fifth and final round began any one of these three contestants was in a position to win the tournament. CHESS CHALLENGER held a half point lead over MIKE and BORIS, but C.C. had to play MIKE in the last round and so a tie for first place was quite possible. While BORIS won its last round game fairly easily, the battle between MIKE and C.C. was very hard fought.

White: CHESS CHALLENGER
Black: MIKE

Queen's Gambit Accepted

1	d2-d4	d7-d5
2	c2-c4	d5xc4
3	e2-e4	Nb8-c6
4	d4-d5	Nc6-e5
5	Qd1-d4	Ne5-d3ch

The best solution. White maintains its pawn centre but Black gets the advantage of two bishops against bishop and knight.

6	Bf1xd3	c4xd3
7	Qd4xd3	c7-c6!

Attacking White's pawn centre.

8	Ng1-e2	Ng8-f6
9	Nb1-c3	Bc8-e6

The right idea but the wrong move. Black could play h7-h6 and then e7-e6, increasing the pressure on d5.

10	Ne2-f4	Be6-d7
11	d5xc6	Bd7xc6
12	Qd3-c4	e7-e5
13	Nf4-d5	Bc6xd5
14	e4xd5	Bf8-d6
15	Qc4-b5ch	Qd8-d7
16	Qb5xd7ch	Nf6xd7

Better is 16 . . . Ke8xd7. The king is needed in the centre for the end game but most chess programs like to castle so much that they do not employ this heuristic.

17	Nc3-b5	Bd6-b4ch
18	Bc1-d2	Bb4xd2ch
19	Ke1xd2	0-0
20	d5-d6	

White does not realise that the further this pawn advances the more difficult it will be to support it.

20	. . .	f7-f5
21	Rh1-c1	Nd7-f6!

Threatening 22 . . . Nf6-e4ch in some positions, forking the king and the pawns on f2 and d6.

22	f2-f3	Ra8-d8
23	Rc1-c7	a7-a6
24	Nb5-c3	Rd8xd6ch
25	Kd2-e3	

Although this was the move displayed by CHESS CHALLENGER the move made on its internal board was different. After

25	. . .	f5-f4ch

CHESS CHALLENGER tried to play 26 Rc7xb7, which is, of course, illegal (since White is in check).

The C.C. operator verified the position and discovered that although d2-e3 had been displayed at move 25 the machine thought that the white king was on e2. After some difficulty the correct position was reset in C.C. and the game continued. It is still not clear whether the error in this game and in the CHESS CHALLENGER'S first round encounter are due to a bug in the machine (the latest, 10 level version) or whether the cause was the fluctuating voltage that had disturbed MIKE in round one.

26	Ke3-f2	Rd6-d2ch
27	Kf2-g1	Rd2xb2
28	Ra1-e1	Rf8-e8
29	Rc7-c5	Rb2-c2
30	Rc5-c7	e5-e4!
31	f3xe4	Re8xe4
32	Rc7-c8ch	Re4-e8??

Why? After 32 . . . Kg8-f7 33 Rc8-c7ch Kf7-g6, White would probably have nothing better than 34 Nc3xe4 Rc2xc7 35 NE4xf6, when Black would have two extra pawns in the ending. Now, however, White can simply take the piece.

33	Re1xe8ch	Nf7xe8
34	Rc8xe8ch	Kg8-f7
35	Re8-c8	

Presumably this is what MIKE missed when playing its 32nd move. The reason is probably that MIKE realised that both the knight and the white rook are under attack, but did not look for a way to defend both of them simultaneously. This, in turn, is probably because its search could not go more than 5-ply when calculating its 32nd move.

35	. . .	h7-h5
36	Rc8-c7ch	Kf7-e6
37	a2-a4	g7-g5
38	h2-h3	b7-b6
39	Rc7-c6ch	

CHESS CHALLENGER can win quite easily by unpinning the knight, e.g. 39 Nc3-d5! Rc2-b2 40 Rc7-c6ch and 41 Rc6xb6. But without being able to unearth this concept it is almost certain to fall prey to the

eventual advance of the black-b-pawn, which will decide the game.

39	. . .	Ke6-e5
40	Rc6-c4	b6-b5
41	a4xb5	Rc2-c1ch
42	Kg1-h2	a6xb5
43	Rc4-c7	b5-b4!
44	Nc3-d5	

Forced, but not it is too late.

44	. . .	Rc1xc7
45	Nd5xc7	b4-b3

The white knight could run back to guard b1 but then Black could win on the king side.

46	g2-g4	b3-b2
47	Kh2-g2	b2-b1=Q
48	Nc7-e8	Qb1-a2ch
49	Kg2-f1	f4-f3!

The quickest way to end the game.

50	Ne8-g7	Qa2-g2ch

and mates next move (51 Kf1-e1 Qg2-e2).

In my opinion this was the most interesting game of the tournament.



Boris *(Rex Kent) playing* Mike *(Mike Johnson)*

MIKE's win produced a tie for first place, as can be seen from the following cross table of the tournament.

	Program	1	2	3	4	5	6	Total	Place
1	MIKE	x	1	1	½	0	1	3½	1–2
2	BORIS	0	x	1	1	½	1	3½	1–2
3	CHESS CHALLENGER	0	0	x	1	1	1	3	3rd
4	MICROCHESS 2.0	½	0	0	x	1	1	2½	4th
5	FAFNER	1	½	0	0	x	½	2	5th
6	COCMA	0	0	0	0	½	x	½	6th

A play-off game took place on the final afternoon of the PCW show in which MIKE and BORIS played with the opposite colours to those that they had during the tournament. Unfortunately this game was marred by yet another strange happening.

White: BORIS
Black: MIKE

Irregular Opening

1	e2-e4	e7-e5
2	d2-d4	d7-d5
3	Nb1-c3	d5xe4
4	d4xe5	Qd8xd1ch
5	Ke1xd1	f7-f6
6	Bc1-e3	f6xe5
7	Kd1-c1?	

I could not understand why BORIS did not capture on e4.

7	. . .	Ng8-f6
8	Bf1-c4	Bf8-d6
9	Ra1-b1	

Another peculiar move.

9	. . .	Nb8-c6
10	Kc1-d1	Bc8-f5
11	Ng1-e2	0-0-0
12	Kd1-c1	

What is all this king shunting about?

12	. . .	Nf6-g4
13	Ne2-g3	Ng4xe3
14	f2xe3	Nc6-a5?!

A positional mistake. The knight is badly placed on the edge of the board.

15	Ng3-f5?	

White should retreat the c4 bishop to e2.

15	. . .	Na5xc4
16	Rh1-f1	g7-g6
17	Kc1-d1??	

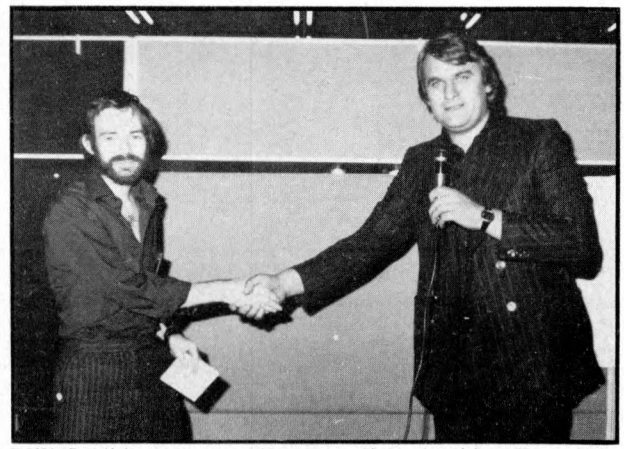
An inexplicable blunder; at least it was inexplicable at the time. After he returned home BORIS's operator, Rex Kent, tried his position several times and not once did BORIS so much as think about moving its king. Possibly a voltage fluctuation was causing the strange king moves. It seems likely that in future events some sort of stabilising system must be used in order to protect the programs.

17	. . .	g6xf5
18	Kd1-e2	Rh8-g8
19	b2-b3	Rg8xg2ch
20	Rf1-f2	Nc4xe3!!

Winning another pawn. If now 21 Ke2xe3 Bd6-c5ch followed by 22 . . . Rg2xf2 wins even more material.

21	Rf2xg2	Ne3xg2
22	Nc3-d5	Ng2-f4ch
23	Nd5xf4	e5xf4

and Black won without any difficulty. First prize was £200, second was 2 bottles of Scotch and a bottle of champagne!



PCW *Publisher presents cheque to Championship winner Mike Johnson*

I was very pleased (if an unbiased arbiter is permitted to be pleased) at the result of this tournament, because it shows that one individual who is programming for a hobby can easily produce a better result than the resources of a commercial company. Had MIKE not had hardware problems in the first round it would have won the tournament with 4½ out of 5, and I think that it must be agreed that the best program won.

I should like to thank PCW for organising the event and for inviting me to direct it. I am beginning to prefer computer chess to competing in international tournaments!

A bulletin containing the moves of all the games is available from Personal Computer World, 62a Westbourne Grove, London W2. Send 40p and a self addressed envelope (9" x 6").

DIODES/ZENERS

1N914	100v	10mA	.05
1N4005	600v	1A	.08
1N4007	1000v	1A	.15
1N4148	75v	10mA	.05
1N4733	5.1v	1 W Zener	.25
1N753A	6.2v	500 mW Zener	.25
1N758A	10v	"	.25
1N759A	12v	"	.25
1N5243	13v	"	.25
1N5244B	14v	"	.25
1N5245B	15v	"	.25

SOCKETS/BRIDGES

8-pin	pcb	.20	ww	.35
14-pin	pcb	.20	ww	.40
16-pin	pcb	.20	ww	.40
18-pin	pcb	.25	ww	.75
22-pin	pcb	.35	ww	.95
24-pin	pcb	.35	ww	.95
28-pin	pcb	.45	ww	1.25
40-pin	pcb	.50	ww	1.25
Molex pins .01		To-3 Sockets		.25
2 Amp Bridge		100-prv		.95
25 Amp Bridge		200-prv		1.95

TRANSISTORS, LEDS, etc.

2N2222	NPN (2N2222 Plastic .10)	.15
2N2907	PNP	.15
2N3906	PNP (Plastic - Unmarked)	.10
2N3904	NPN (Plastic - Unmarked)	.10
2N3054	NPN	.35
2N3055	NPN 15A 60v	.50
T1P125	PNP Darlington	.35
LED Green, Red, Clear, Yellow		.15
D.L.747	7 seg 5/8" High com-anode	1.95
MAN72	7 seg com-anode (Red)	1.25
MAN3610	7 seg com-anode (Orange)	1.25
MAN82A	7 seg com-anode (Yellow)	1.25
MAN74A	7 seg com-cathode (Red)	1.50
FND359	7 seg com-cathode (Red)	1.25

C MOS

4000	.15
4001	.15
4002	.20
4004	3.95
4006	.95
4007	.20
4008	.75
4009	.35
4010	.35
4011	.20
4012	.20
4013	.40
4014	.75
4015	.75
4016	.35
4017	.75
4018	.75
4019	.35
4020	.85
4021	.75
4022	.75
4023	.20
4024	.75
4025	.20
4026	1.95
4027	.35
4028	.75
4030	.35
4033	1.50
4034	2.45
4035	.75
4040	.75
4041	.69
4042	.65
4043	.50
4044	.65
4046	1.25
4049	.45
4050	.45
4066	.55
4069/74C04	.25
4071	.25
4081	.30
4082	.30
MC 14409	14.50
MC 14419	4.85
4511	.95
74C151	1.90

– T T L –

7400	.10	7473	.25	74176	.85	74H72	.35
7401	.15	7474	.30	74180	.55	74H101	.75
7402	.15	7475	.35	74181	2.25	74H103	.55
7403	.15	7476	.40	74182	.75	74H106	.95
7404	.10	7480	.55	74190	1.25		
7405	.25	7481	.75	74191	.95	74L00	.25
7406	.25	7483	.75	74192	.75	74L02	.20
7407	.55	7485	.55	74193	.85	74L03	.25
7408	.15	7486	.25	74194	.95	74L04	.30
7409	.15	7489	1.05	74195	.95	74L10	.20
7410	.15	7490	.45	74196	.95	74L20	.35
7411	.25	7491	.70	74197	.95	74L30	.45
7412	.25	7492	.45	74198	1.45	74L47	1.95
7413	.25	7493	.35	74221	1.00	74L51	.45
7414	.75	7494	.75	74367	.75	74L55	.65
7416	.25	7495	.60			74L72	.45
7417	.40	7496	.80	75108A	.35	74L73	.40
7420	.15	74100	1.15	75491	.50	74L74	.45
7426	.25	74107	.25	75492	.50	74L75	.55
7427	.25	74121	.35			74L93	.55
7430	.15	74122	.55			74L123	.85
7432	.20	74123	.35	74H00	.15		
7437	.20	74125	.45	74H01	.20	74S00	.35
7438	.20	74126	.35	74H04	.20	74S02	.35
7440	.20	74132	.75	74H05	.20	74S03	.25
7441	1.15	74141	.90	74H08	.35	74S04	.25
7442	.45	74150	.85	74H10	.35	74S05	.35
7443	.45	74151	.65	74H11	.25	74S08	.35
7444	.45	74153	.75	74H15	.45	74S10	.35
7445	.65	74154	.95	74H20	.25	74S11	.35
7446	.70	74156	.70	74H21	.25	74S20	.25
7447	.70	74157	.65	74H22	.40	74S40	.20
7448	.50	74161	.55	74H30	.20	74S50	.20
7450	.25	74163	.85	74H40	.25	74S51	.25
7451	.25	74164	.60	74H50	.25	74S64	.15
7453	.20	74165	1.10	74H51	.25	74S74	.35
7454	.25	74166	1.25	74H52	.15	74S112	.60
7460	.40	74175	.80	74H53J	.25	74S114	.65
7470	.45			74H55	.20		
7472	.40						

74S133	.40
74S140	.55
74S151	.30
74S153	.35
74S157	.75
74S158	.30
74S194	1.05
74S257 (8123)	1.05
74LS00	.20
74LS01	.20
74LS02	.20
74LS04	.20
74LS05	.25
74LS08	.25
74LS09	.25
74LS10	.25
74LS11	.25
74LS20	.20
74LS21	.25
74LS22	.25
74LS32	.25
74LS37	.25
74LS38	.35
74LS40	.30
74LS42	.65
74LS51	.35
74LS74	.35
74LS86	.35
74LS90	.55
74LS93	.55
74LS107	.40
74LS123	1.00
74LS151	.75
74LS153	.75
74LS157	.75
74LS164	1.00
74LS193	.95
74LS367	.75
74LS368	.65

9000 SERIES

9301	.85	95H03	1.10
9309	.35	9601	.20
9322	.65	9602	.45

LINEARS, REGULATORS, etc.

MCT2	.95	LM320T5	1.65	LM340K15	1.25	LM723	.40
8038	3.95	LM320T12	1.65	LM340K18	1.25	LM725N	2.50
LM201	.75	LM320T15	1.65	LM340K24	1.25	LM739	1.50
LM301	.45	LM324N	1.25	78L05	.75	LM741 (8-14)	.25
LM308 (Mini)	.95	LM339	.75	78L12	.75	LM747	1.10
LM309H	.65	7805 (340T5)	.95	78L15	.75	LM1307	1.25
LM309K (340K-5)	.85	LM340T12	.95	78M05	.75	LM1458	.65
LM310	.85	LM340T15	.95	LM373	2.95	LM3900	.50
LM311D (Mini)	.75	LM340T18	.95	LM380 (8-14 PIN)	.95	LM75451	.65
LM318 (Mini)	1.75	LM340T24	.95	LM709 (8, 14 PIN)	.25	NE555	.35
LM320K5 (7905)	1.65	LM340K12	1.25	LM711	.45	NE556	.85
LM320K12	1.65					NE565	.95
						NE566	1.25
						NE567	.95

MICRO'S, RAMS, CPU'S, E-PROMS

74S188	3.00	8214	8.95
1702A	4.50	8224	3.25
MM5314	3.00	8228	6.00
MM5316	3.50	8251	8.50
2102-1	1.45	8255	10.50
2102L-1	1.75	8T13	1.50
2114	9.50	8T23	1.50
TR1602B	3.95	8T24	2.00
TMS 4044-	9.95	8T97	1.00
		2107B-4	4.95
8080	8.95	2708	9.50
8212	2.95	Z80 PIO	8.50

INTEGRATED CIRCUITS UNLIMITED

7889 Clairemont Mesa Blvd., San Diego, CA 92111 U.S.A.

No Minimum

All prices in U.S. dollars. Please add postage to cover method of shipping. Orders over $100 (U.S.) will be shipped air no charge. Payment should be submitted with order in U.S. dollars. All IC's Prime/Guaranteed. All orders shipped same day received.

SPECIAL DISCOUNTS

Total Order	Deduct
$35 - $99	10%
$100 - $300	15%
$301 - $1000	20%

Phone (714) 278-4394 **BarclayCard / Access / American Express / BankAmericard / Visa / MasterCharge**

AN AGILE ALGORITHM

Square root by addition

R. E. L. Ferguson

A simple program for your micro which requires only addition for its operation.

The program makes use of the well known mathematical relationship,

$$\sum_{1}^{n} (2n - 1) = n^2,$$

which means, in effect, "If the sum of successive odd numbers is taken, the total at any point in the series will be the square of the number of terms".

e.g. $1 + 3 = 4 = 2^2$

$1 + 3 + 5 = 9 = 3^2$

$1 + 3 + 5 + 7 = 16 = 4^2$, ... etc.

Considering this in reverse, the number of terms serves as a means of evaluating the square root of the sum.

For program use, four storage locations are required; one to hold the number (N) whose root is sought, one to store the current sum of the odd numbers, one to hold successive odd numbers and a 'counting' store to record the number of additions, this providing the required square root. As each new odd number is added, the current sum is compared with N, the operation being stopped when this value is equalled (or exceeded).

All of this is very straightforward when dealing with perfect squares, but what about $\sqrt{7}$, $\sqrt{123}$, etc.? For these we adopt the system familiar to users of square root tables. If a number lies outside the range of the tables, it is increased or decreased by an even power of ten, as follows:

$\sqrt{1234} = \sqrt{(12.34 * 100)} = 3.513 * 10 = 35.13$, or,

$\sqrt{0.01234} = \sqrt{(1.234 \div 100)} = 1.111 \div 10 = 0.1111$

Similarly, by using a location with two (or four or six . . .) *additional* decimal digits when we sum the odd numbers, we will, in effect, be multiplying by 10^2 (or 10^4 or 10^6 . . .).

For example, $\sqrt{7}$ becomes $\sqrt{700}$ (or $\sqrt{70000}$, or $\sqrt{7000000}$ . . .) which requires 27 additions (or 265, or 2646 . . .). Thus $\sqrt{7} = 2.7$ (or 2.65, or 2.646 . . .) More accuracy can be obtained by extending still further the number of digits (always in pairs, of course, to obtain an even power of ten).

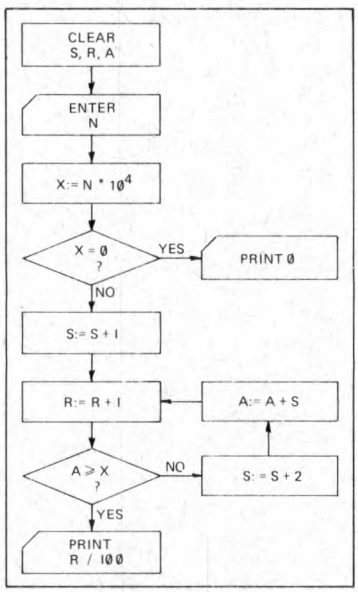


Fig. 1

Obviously there is a catch! Limitations on accuracy are set by a time factor. As the size of the store increases, so does the time required to fill it and this becomes excessive for anything over six digits. At five-digit level, $\sqrt{99999}$ takes less than 1.5 secs, but the largest number with six digits, $\sqrt{999999}$ requires about ten secs! (for the 6800 at 1 MHz). If this is regarded as just about acceptable for the home-computer enthusiast, the program is worth considering.

Integers up to six digits can be dealt with and the result given to two decimal places, e.g. $\sqrt{123456}$ = 351.37 (The last figure may, on occasion, be inaccurate by +1)

The program uses locations $00F4_{16}$ to $01D7_{16}$ but is easily relocated.

The flow chart could be as follows: (see Fig. 1)

N = number to be entered
S = store holding successive odd numbers
R = result (counting) store
A = addition store for summing odd numbers

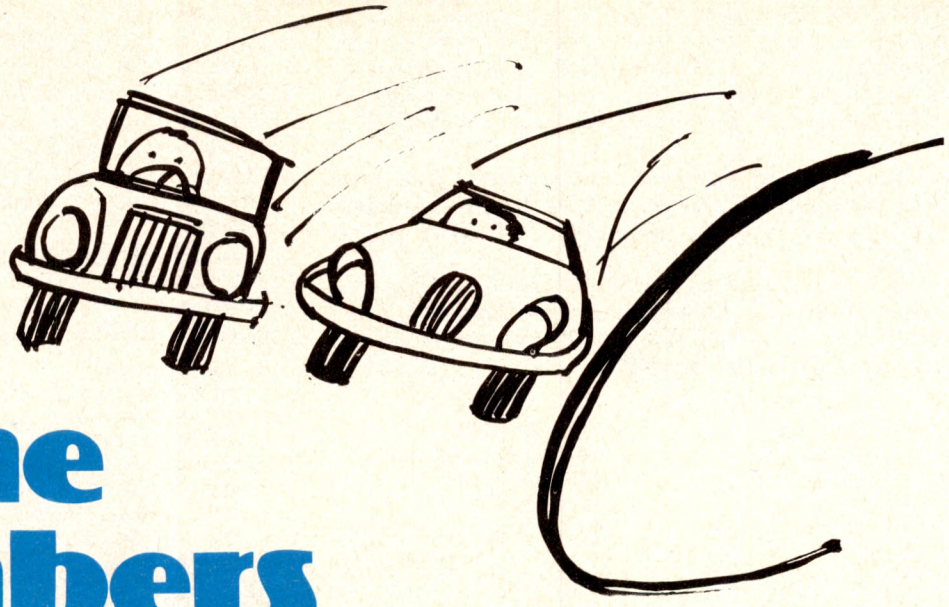
The program was developed for use with the MIKBUG (6800) system, but could be re-written for any other instruction set.

It starts at 0100, which should be entered at A048, A049 in the usual way for MIKBUG, followed by 'G', for 'go to user's program'. If the starting address is changed, certain other locations will also have to be altered. These are underlined in the listing. ALL others are relative and need not be moved. The program expects a six-digit number, so less than six digits must be entered with leading zeros; i.e. 123 becomes 000123. After each result, the program is restored to MIKBUG and waits for 'G', which returns it to 0100 for the next number. If required as a sub-routine, clear locations 0016 to 0022, load 0010, 11 and 12 with the six digits of the number and change the contents of 01B5 to 39 (RTS). The remainder of the program (01B6 to 01D7) can be discarded and the result will be held in 0020, 21 and 22.

SIX-DIGIT SQUARE ROOT

```
00F4  0A  0D20          LF, CR,
  F7  20  2020          SPACES
  FA  20  2020
  FD  20  2004

0100  CE  0010  LDX, #  START
0103  6F    00  CLR, X
   5  08    --  INX
   6  8C  0025  CPX, #
   9  27    03  BEQ      0120
   B  7E  0103  JMP
   E  BD  E055  JSR      INPUT N1
0111  97    10  STA A    N1 → 10
  13  BD  E055  JSR      INPUT N2
  16  97    11  STA A    N2 → 11
  18  BD  E055  JSR      INPUT N3
  1B  97    12  STA A    N3 → 12
  1D  26    0A  BNE      0129
  1F  96    11  LDA A
0121  26    06  BNE      0129
  23  96    10  LDA A
  25  26    02  BNE      0129
  27  20    42  BRA      016B   (PRINT 0)
  29  7C  001F  INC
  2C  20    22  BRA      0150

012E  96    1F  LDA A                    INCREMENT ODD NUMBER TO NEXT
  30  8B    02  ADD A, #
  32  19    --  DAA
  33  97    1F  STA A
  35  24    19  BCC      0150
  37  96    1E  LDA A
  39  8B    01  ADD A, #
  3B  19    --  DAA
  3C  97    1E  STA A
  3E  24    10  BCC      0150
0140  96    1D  LDA A
  42  8B    01  ADD A, #
  44  19    --  DAA
  45  97    1D  STA A
  47  24    07  BCC      0150
  49  96    1C  LDA A
  4B  8B    01  ADD A, #
  4D  19    --  DAA
  4E  97    1C  STA A

0150  96    22  LDA A                   INCREMENT RESULT
  52  8B    01  ADD A, #
  54  19    --  DAA
  55  97    22  STA A
  57  24    12  BCC      016B
  59  96    21  LDA A
  5B  8B    01  ADD A, #
  5D  19    --  DAA
  5E  97    21  STA A
0160  24    09  BCC      016B
  62  96    20  LDA A
  64  8B    01  ADD A, #
  66  19    --  DAA
  67  97    20  STA A
  69  01    01  NOP

016B  96    1F  LDA A                   ADD NEXT ODD NUMBER
  6D  9B    1A  ADD A
  6F  19    --  DAA
0170  97    1A  STA A
  72  96    1E  LDA A
  74  99    19  ADC A
  76  19    --  DAA
  77  97    19  STA A
  79  96    1D  LDA A
  7B  99    18  ADC A
  7D  19    --  DAA
  7E  97    18  STA A
0180  96    1C  LDA A
  82  99    17  ADC A
  84  19    --  DAA
  85  97    17  STA A
  87  24    0B  BCC      0194
  89  96    16  LDA A
  8B  8B    01  ADD A, #
  8D  19    --  DAA
  8E  97    16  STA A
0190  24    02  BCC      0194
  92  20    21  BRA      01B5

0194  D6    10  LDA B                   TEST
  96  D1    16  CMP B
  98  22    18  BHI      01B2
  9A  27    02  BEQ      019E
  9C  20    17  BRA      01B5
  9E  D6    11  LDA B
01A0  D1    17  CMP B
  A2  22    0E  BHI      01B2
  A4  27    02  BEQ      01A8
  A6  20    0D  BRA      01B5
  A8  D6    12  LDA B
  AA  D1    18  CMP B
  AC  22    04  BHI      01B2
  AE  27    05  BEQ      01B5
01B0  20    03  BRA      01B5
  B2  7E  012E  JMP      REPEAT

01B5  CE  00F4  LDX                    PRINT RESULT
  B8  BD  E07E  JSR
  BB  CE  0020  LDX, #
  BE  BD  E0BF  JSR
01C1  CE  0021  LDX, #
  C4  BD  E0BF  JSR
  C7  86    2E  LDA A, #
  C9  BD  E1D1  JSR
  CC  CE  0022  LDX, #
  CF  BD  E0BF  JSR
01D2  BD  E0E3  JSR
  D5  7E  0100  JMP      TO START
```

To use full five-digit accuracy for small numbers, enter with the value increased by a factor of 10^2 (or 10^4).
For example, 5 is entered as 50000, giving 223.61, to be interpreted as 2.2361.



Prime Numbers Competition

Sheridan Williams

The July edition of PCW included an article entitled "Faster, More Efficient Programs". At the end of that article was a competition, which was as follows: Write two programs to fill the array A(1) to A(100) with the first 100 prime numbers; the first program is to work the fastest, and the second is to contain the fewest number of statements. One month from the date of publication was allowed for the entries. After that time detailed trials were undertaken to ascertain the speeds of the various programs.

A prime number is defined as:– a number that is only divisible by itself and unity. One is not considered to be a prime. The lowest prime, and the only even prime, is two. Prime numbers have always fascinated man, and since the advent of computers a great deal of computer time has been spent on finding the highest prime number. It can be shown that the number of primes is infinite, therefore there is no highest prime. We can always find a higher prime by multiplying all the primes found so far and adding one. Also note that primes are of the form $6n \pm 1$ (ie not multiples of 2 and 3). No one has yet found a formula that will generate prime numbers; there are formulae that work for a while, a simple example is $n^2 + n + 41$, try substituting n = 1,2,3.... you get 43, 47, 53, 61... all of which are primes. This formula breaks down after a while, can you spot where?

The fellow who is reputed to have started the work on prime numbers was born in 270 BC, and was called Eratosthenes. He lived and taught in a school in Alexandria, and died in 190 BC. He developed a method for finding prime numbers known thereafter as the "Sieve of Eratosthenes". The method will be explained later.

I received a great many more entries than I had anticipated. The result was that the standing of the winning entries was much higher. I am still very interested, (and I am sure many readers would be too) to hear from any person who feels that he can better either of the winning programs. The entries that I received came from all over England, Scotland, Wales, N. Ireland, Eire, and even from Norway. Programs were submitted by all ages from 12 years upwards. It is notable, though that only *one* entry came from a girl, and that girl was aged 12. Let's hope that we get more entries from the fair sex next time, after all women are better programmers than men, aren't they? (The storm of letters that will follow to **PCW** not me, please).

Most programs were written in BASIC, which was what I had intended. However I did receive two written in ALGOL, my favourite language (I do not know APL well enough to comment), and streets ahead of FORTRAN. (More letters, all to **PCW** not me. The idea of a magazine such as **PCW** is to provoke controversy in the letters page – I'm just doing my bit). It was very difficult to judge the two ALGOL programs, because they needed to be translated into BASIC to be fair to all the other entrants. It would have been unfair to time the programs without translation, because ALGOL would have been compiled and not interpreted, and hence would have been faster. One of the programs was so advanced that its translation into BASIC would have been impossible.

Over the period of the competition I was extremely lucky to have two micro-computers for evaluation. The first was the already familiar Commodore PET, which needs no further description because it has already been evaluated briefly, in issue 2 by John Coll. The second computer was a Cromemco Z–2D on loan from **Comart** of St. Neots. A review of this machine is to follow in a future issue. I will give a summary of the bench mark tests so that the program speeds listed in this article can be compared with the same programs run on other computers. I have given the figures for the PET in case you missed them in a previous issue.

Test	PET	Z–2D	
BM1	1.5	2.0	
BM2	9.7	5.8	
BM3	18.2	16.4	
BM4	20.3	19.1	
BM5	21.8	20.1	
BM6	32.4	31.5	
BM7	50.9	42.5	
BM8	12.2	23.0	(all times in seconds)
BM7+8	63.1	65.5	

The Z–2D was used for the initial testing, because having a floppy disc system, I could store all the programs submitted on the disc, and recall any one within 2 seconds for comparison. I was loaned the Z–2D for only one week, so final timings were done on the PET.

The programs submitted can be divided into three clear categories:– 1. Trivial. 2. Mathematical (successive divisions and tests for remainder). 3. Sieve of Eratosthenes.

Examples of category ONE are given first. They are a result of my not stating the problem properly, and I accept full responsibility for that omission. There are two types within this category:

TYPE 1

```
10 DIM A(100)
20 LET A(1) = 2
30 LET A(2) = 3
40 LET A(3) = 5
50 LET A(4) = 7

60 etc
```

TYPE 2

```
10 DIM A(100)
20 FOR X = 1 TO 100
30 READ A(X)
40 NEXT X
50 DATA 2,3,5,7,11,13,17...
                        etc
60 DATA etc
```

These two programs were rejected for two reasons. Firstly they depended on knowing all 100 primes before the programs were written, and hence the programs could not easily be extended to find 1000 primes. Secondly, they were submitted by the majority of entrants (most entrants realised my error and also submitted sensible programs). On that basis I would have had to award a prize of a few pence each, being unable to find a clear winner.

Category Two provided the winner of the section program with the "fewest statements", these programs were written with total disregard for speed, as intended. The winning program came from **David Gutteridge** of Bromsgrove, it is listed below together with some of his comments:

```
10 DIM A(100)
20 FOR K = 1 TO 100
30  LET C = C + 1 + ABS(SGN(C–2) )
40   FOR J = 2 TO SQR(C + 1)
50    IF C/J = INT(C/J) * SGN(C–2) THEN 30
60   NEXT J
70   A(K) = C
80 NEXT K
90 END
```

Line 30: the expression ABS(SGN(C–2)) returns 1 in all cases except when C = 2 when it returns zero. Successive visits to line 30 produce values for C = 2,3,5,7,9,11, 13... ie the candidates examined as primes are 2 and then all odd numbers from 3 upwards.
Line 40: Only need to test factors between 2 and √C, however many interpreters give SQR(9) = 2.9999999, taking SQR(C + 1) as an upper limit ensures that 3 is picked as a factor of 9, 5 as a factor of 25 etc.
Line 50: As mentioned later on in this article many interpreters perform the FOR loop once, even if the second parameter is less than the first. Thus line 50 has to contain this dodge to cause failure of the decision when C = 2.

This program is one line shorter than two others that I received and hence is the outright winner. Category TWO has one main reason why it failed to supply the "fastest" program, that reason is "arithmetic". The method requires successive divisions of the number being tested for primeness by all the odd digits up to its square root (even numbers need not be tested if 2 is put into A(1) before the program starts), the result is then tested for being an integer, if no division results in an integer then the number must be prime. Let us look at the number of calculations performed to test whether 541 is prime; this requires a minimum of 8 divisions, 8 arithmetic tests, and 8 calls of the INT function to determine that it *is* prime; similarly 437 requires almost as many divisions, tests, and INT calls to determine that it is *not* prime. Using this method it takes successively longer to find each prime, and cannot be recommended for finding very large primes. If the primes are not stored when they are found, this method does have the advantage of not requiring the store space that category THREE requires.

The fastest program in this category, (TWO), although not fast enough for the prize, came from W. S. Lounds of Lancaster. It ran in 19½s. With the modifications listed I managed to reduce the time to 17s.

```
 10 DIM A(100)
 20 LET N = 2
 30 LET A(1) = 2
 40 LET A(2) = 3
 50 FOR J = 5 TO 1001 STEP 2
 60  FOR Q = 2 TO N
 70   IF A(Q) * A(Q) > J THEN 100
 80   IF J/A(Q) = INT(J/A(Q) ) THEN 130
 90  NEXT Q
100  LET N = N + 1
110  LET A(N) = J
120  IF N = 100 THEN 140
130 NEXT J
140 END
```

The following amendments give a 2½s improvement:

```
65 LET T = J/A(Q)
70 IF A(Q) > T THEN 100
80 IF T = INT(T) THEN 130
```

Other notable programs in this group came from **J. McCosh** of Crossabeg, Co. Wexford, Ireland; Miss **J. M. Broughton**, Northwood, Middx; **D. C. Broughton**, Northwood, Middx; **S. Dorritt**, Chalfont St Giles; **S. Morris**, Bodenham.

Category THREE provided the fastest program, which ran in the amazing time of 5½s. This category used a technique called the "sieve of Eratosthenes". The basic technique is illustrated below:

1) Write out the integers from 2 upwards.
2) Circle the 2 and cross off all multiples of 2 upwards.
3) Starting from the left move along the list until you come to a number that hasn't been crossed off.
4) Circle this number, and cross off all multiples above it.
5) Go back to step (3) until 100 numbers have been circled.

That is the technique, although there are ways of making it faster. Only three programs of this type were submitted, by **R. Ferguson** of Brighton, **A. K. Booer** of Oxford, and **R. Furse** of Leamington Spa.

The winning program was by **A. K. Booer** who used some very clever techniques which reduced storage requirements and decreased running time by half. This program is given below, together with his comments on its operation:

```
 10 DIM A(100) ,B(270)
 16 LET M = 0
 18 LET A(1) = 2
 20 FOR J = 2 TO 100
 30   LET M = M + 1
 35   IF B(M) = 1 THEN 30
 40   LET N = M + M + 1
 45   LET A(J) = N
 50   FOR X = M TO 270  STEP N
 54     LET B(X) = 1
 57   NEXT X
 60 NEXT J
 70 END
```

NOTES: Only odd numbers are tested for being prime, the Mth element of B represents the number 2 * M + 1. As primes are found their (odd) multiples are flagged by setting the corresponding elements in array B to 1, up to the limit of 541.

16 initialises the pointer to B array
18 sets A(1) to the only even prime (since only odd numbers tested)
20 counts the number of primes found.
30 increments M pointer to B array
35 checks to see if element in B being flagged is a multiple of a previous prime, loops back to 30 if this is so.
40 N is the next prime, so store it in A
50-53 flags odd multiples of N as being non-prime
60 loops back for the next prime.

The efficiency of this program lies in the fact that there are no multiplications, divisions, or function evaluations, and only one test (excluding FOR..NEXT loops).

The above program runs in 5½s and can be made minutely faster by the following amendments:

```
47  LET P = N*N/2-1
50  FOR X = P TO 270 STEP N
```

As an exercise perhaps you can see why.

Bearing in mind the time for the above program, perhaps I could issue an open challenge to anyone to beat it. The competition is as follows. The program is to be in BASIC and is to find the first 1000 primes (this makes timing easier and more accurate). Please include with all entries; the listing, a copy of the output, and the run time mentioning the computer and interpreter used; also include the run time for the above program (extended to work for 1000 primes) for comparison purposes. Send entries together with a SAE to me at *114, Beech Road, St Albans, Herts, AL3 5AU.*

I would like to thank all those who took the trouble to enter the competition, and apologise for not being able to mention every name, I saw some very creditable entries and wish them luck with future competitions.

The judging of the competition was not without its problems, I would like to mention one in particular which I came across, and one which I feel programmers should be aware of. It concerns the operation of the FOR...NEXT loop. Suppose we have a loop as follows 10 FOR X = A TO B 20 NEXT X. Some BASIC interpreters work differently from others; the poorer (extremely bad) interpreters perform the loop check at the NEXT statement instead of at the FOR statement. This means that if B is less than A the loop is performed once before a check is made. Some of the competition entries required an additional line to overcome this. To check which kind of interpreter you have, try running this program; if the output is as predicted, then you have one of the bad interpreters, the FOR statement will have to be preceded by the suggested line:

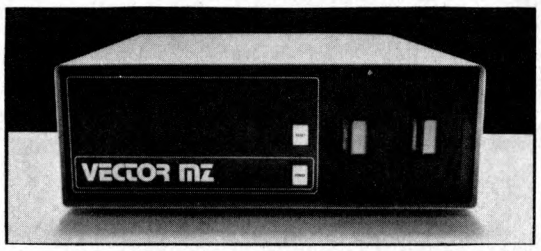
```
10 LET A = 10
20 LET B = 5
30 FOR X = A TO B
40   PRINT "X=";X
50 NEXT X
60 END
```

To overcome BAD interpreters this line must be inserted:—

```
25 IF B<A THEN 60
```

Predicted output for a BAD interpreter is:— X = 10

The next competition is more demanding and requires you to write a BASIC program to print 4 bridge hands. There are two categories (i) the fastest; (ii) the fewest statements (multi-statement lines will be disallowed to enable everyone to compete on equal terms). No trivial entries please. The output required is the card value and suit, example 4H,10C,QD,AS,8S, and each hand labelled NORTH SOUTH EAST WEST. Each run of the program should give different hands. **A prize of £10 for each category** will be awarded. Entries must be posted to the address in this article, including SAE, by 15th of the month following publication; please state clearly which category you are entering. Include the program listing and sample output.



THE TOTAL SOLUTION FROM

ALMARC

OF COURSE!

Now Almarc & Vector Graphic offer the complete solution to your computing needs for £2300.00*. The Vector MZ needs only the addition of a V.C.U. and its ready to go. Completely assembled and fully tested the Vector MZ offers the following features as standard:—

* S-100 bus
* 4MHz Z80 processor
* 158 instructions
* two quad density Micropolis floppies — over 630K bytes on line
* serial port
* two parallel ports
* 32K static ram
* 12K prom/ram board with extended monitor
* Extended disc Basic

Simply connect your peripherals (Elbit V.D.us & Centronics printers are available from Almarc) and your up and running and, because the MZ uses the S-100 bus, you can plug in a massive range of add on units.

Ring or Write for a demonstration to:-
Almarc Data Systems Ltd.,
29 Chesterfield Drive, Burton Joyce, Nottingham.
Telephone 0602 248565

***Discount terms available**

EQUINOX 300

A powerful multi-user
multi-tasking
multi-language

16-bit microcomputer time-sharing system

supporting
* BASIC
* LISP
* PASCAL
* Floppy Discs
* Hard discs

including a powerful Text Formatter, Assembly Language Development System and disc based Sort utilities.

Priced from under £5,000

Write or phone for further information

EQUINOX COMPUTER SYSTEMS LTD.
32-35 Featherstone Street,
London, EC1Y 8QX
(01-253 3781/9837)





VISIT OUR NEW STORE AT:
2 Gatley Road, Cheadle,
Cheshire. Tel. 061-491 0134

THE NEWBEAR COMPUTING STORE

Send for an up-to-date catalogue to:

The Newbear Computing Store
7 Bone Lane, Newbury.

24 HOUR TURNROUND ON ORDERS

Please add 8% VAT to all prices. P & P 30p unless otherwise stated. Barclaycard & Access welcome. Overseas orders issued with Pro-Forma Invoice. Send for catalogue to **NewBear Computing Store, Bone Lane, Newbury, Berks.** Callers welcome Mon – Sat, 9.00 – 5.30 but please phone us first on 0635 49223. New office – 2 Gatley Road, Cheadle, Cheshire (callers only) Tel: 061 491 0134

Hardware Components Section

SYM 1
Formally VIM-1)

The new 6502 Based Micro from Synertek fully assembled and tested. Send or phone for further details. £199.00
+ 8% VAT Carriage £1.00

BEARBAGS (KITS)

1.	77-68 CPU PCB and components	£ 49.50
2.	77-68 LED's and switches	£ 14.95
3.	77-68 Power supply	£ 17.95
4.	77-68 19" 5u Rack and Backplane	£ 26.70†
5.	77-68 4K Ram PCB and components	£ 75.00
6.	77-68 Mon 1 PCB and components	£ 50.00
7.	4K Ram Exorcisor PCB and components	£ 71.50
8.	8K Ram Exorcisor PCB and components	£160.00*
9.	Petitevid VDU Kit	£ 85.00
10.	Kansas City Cassette interface	£ 18.95
11.	UHF Modulator	£ 4.50
12.	77-68 VDU PCB and components	£ 69.50
15.	PROMVERTER (Enables a 2708 to be used instead of MIKBUG)	£ 8.50

50p p+p unless otherwise stated † 1.50 p+p * 1.00 p+p

Unencoded Keyboard		£27.50 + £1.50 p+p
Licon Assembled & Tested Keyboard		£86.00 + £1.50 p+p
Hexadecimal Keypad (19 positions)		£11.25
Keyboard case		£20.33 + £1.50 p+p
19" 5u Rack		£23.40 + £1.50 p+p
U.V. Prom Eraser		£56.00*
9368 7 seg. Display Decoder		£2.45
FWD 500 ½" 7 seg. Display		£1.05
C-30 Cassette Memorex £0.75	5" Floppy Disc	£4.00
C-60 Cassette Memorex £0.90	8" Floppy Disc	£4.00
C-90 Cassette Memorex £1.20		

*£1.00 Postage & Packing

S-100 BUS

CROMEMCO Z-2 kit (CPU + Card Frame)	£395.00
BYTESAVER PROGRAMMING BOARD kit (2708)	£ 95.00
TU-ART Interface kit	£130.00
16K BYTE Dynabyte ram card (fully tested)	£415.00
North Star Mini Floppy kit	£490.00

MEMORIES

2101–1	£ 1.25
2102L–1	£ 1.35
2112	£ 3.04
2513	£ 6.50
SWATBUG	£16.00
4027	£ 4.50
2114	£ 8.25
2708	£ 6.99
MC6830L7	£13.65
MIKBUG	
4116 16K Dynamic Rams	£15.00

INTERFACING I.C's

SFF 96364	£16.20
MC1488P	£ 1.40
MC1489P	£ 1.40
75150P	£ 1.30
75150N	£ 1.20
75134	£ 2.30
4N33	£ 1.95
AY-5-1013	£ 4.50
6402	£ 4.50

MOTOROLA MICROCOMPUTING I.C.'s

MC6800	£9.26
MC6820	£4.63
MC6850	£6.74
MC6810AP	£3.61
MC8602P	£2.88
MC14536P	£3.69
MC3459	£2.53

ZILOG MICROCOMPUTING I.C's

Z80 CPU 2.5Mhz	£15.50
Z80 PIO	£10.00
Z80 CTC	£10.00
Z80A CPU 4 Mhz	£20.50
Z80A PIO	£11.55
Z80A CTC	£11.55

MICROPROCESSORS

SC/MP 11	£10.30
INS 8154N	£ 8.18
6502	£14.93
8080	£ 6.00

BOOKS

All books are subject to availability

		P. & P.
COMPUTER DESIGNS		
77-68 6800 Microcomputer	£ 7.50	.50
Spare diagram set for 77-68	£ 1.50	.50
WB–1 TTL Microcomputer	£ 6.50	.50
Spare diagram set for WB–1	£ 1.00	.50
FROM ADAM OSBOURNE ASSOCIATES		
Introduction to Microcomputers		
Volume 0: The Beginners Book	£ 5.95	.50
Volume 1: Basic Concepts	£ 5.95	.50
Volume 2: Some Real Products June 1977 Revision	£11.95	1.00
8080A/8085 Assembly Language Programming	£ 6.95	.50
6800 Assembly Language Programming	£ 5.95	.50
Some Common Basic Programs	£ 5.95	.50
6800 Programming for Logic Design	£ 5.95	.50
8080 Programming for Logic Design	£ 5.95	.50
Payroll with Cost Accounting in Basic	£ 9.95	1.00
SCELBI		
Understanding Microcomputers & Small Computer Systems	£ 7.56	.50
Scelbi 6800 Software Gourmet Guide & Cook Book	£ 7.95	.50
Scelbi 8080 Software Gourmet Guide & Cook Book	£ 7.95	.50
8080 Standard Assembler	£15.95	.75
The Scelbi Byte Primer	£ 9.95	1.00
The 8080 Programmers Pocket Guide	£ 2.35	.30
ZILOG		
Z80 Technical Manual	£ 4.00	.50
Z80 PIO Technical Manual	£22.50	.50
Z80 Programming Manual	£ 4.00	.75
MOTOROLA		
Understanding Microprocessors	£ 2.75	.30
M6800 Microprocessor Programming Manual	£ 4.50	.50
M6800 Microprocessor Applications Manual	£ 9.50	1.00
SYBEX		
Microprocessors C201	£ 8.00	.50
Microprocessors Interfacing Techniques	£ 8.00	.50
BASIC		
Instant Freeze-Dried Computer Programming in Basic By Jerald R. Brown	£ 4.95	.75
My Computer Likes Me When I Speak in Basic	£ 1.65	.30
Computer Programs That Work	£ 2.40	.75
Basic Software Library Volume 1:	£17.50	.50
Volume 2:	£17.50	.50
Volume 3:	£26.95	.50
Volume 4:	£ 7.95	.50
Volume 5:	£ 7.95	.50
Volume 6:	T.B.A.	.50
Volume 7:	£26.95	.50
What to do after you hit Return	£ 7.00	.75
NEW BOOKS		
First Book of KIM	£ 7.00	.50
Z80 Microcomputer Handbook	£ 7.25	.50
Using the 6800 Microprocessor	£ 6.25	.50
Micro 6502 Journal	£ 1.70	.50
Learning Basic Fast	£ 6.30	.50
How to Program Microcomputers	£ 6.30	.75
Getting Involved with your own Computer	£ 4.75	.50
8080A Bugbook	£ 6.95	.75
Microcomputer Primer	£5.55	.50
Your Home Computer	£4.95	.50
TTL Cookbook	£6.30	.50
IC Timer Cookbook	£6.95	.75
IC OP AMP Cookbook	£9.05	.75
The Home Computer Revolution	£3.45	.50
Computer Lib	£5.95	.50
Dilithium		
Beginning Basic	£4.95	.50
Home Computers Vol. 1 Hardware	£5.95	.50
Home Computers Vol. 2 Software	£5.95	.50
Home Computers. A beginners Glossary and guide	£4.95	.50
A step by step Introduction to 8080 Microprocessors Systems	£5.70	.50
Take a chance with your calculator	£4.95	.50

A Way With Waves

Gordon Bell

The versatile graphics facilities of the PET can be used to illustrate some basic mathematical functions.

In BASIC, it is easy to calculate the SIN of any angle (expressed in radians) by a statement of the form;-

```
10 LET Y = SIN(X)
```

This statement is incorporated into a BASIC programme and the value obtained for Y is used to calculate the print position in a line, where a "S" character is to be printed. If the program is then caused to loop, varying the value of X from O to 2π (ie. from 0° to 360°, there being 2π radians in 360°), a visual representation of a SIN wave will be portrayed on the VDU screen. The program to do this is as follows:-

```
10 FOR X = O TO 2*π STEP 0.2
20 PRINT TAB(20+(20*SIN(X)));"S"
30 NEXT X
```

If the line below is now added to the program, a COSINE and SIN wave will be illustrated at the same time. This demonstrates visibly the 90 phase shift that exists between these two wave forms.

```
25 PRINT TAB(20+(20*COS(X)));"c"
```

Finally, for those wishing to experiment, some interesting paths can be explored by adding the following line and by varying the character that is to be printed at the chosen location.

```
25 PRINT TAB(20+(20*COS(X+π/2))); "S"
```


INTERESTED IN HOME COMPUTING?

nm Nascom Microcomputers

Start now and don't get left behind

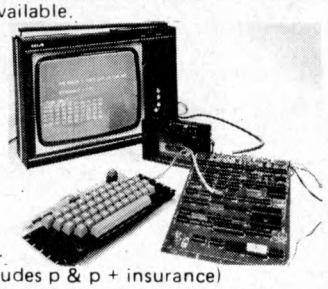
THE NASCOM 1 is here

Ex-stock with full technical services

BLANK C12 Racal Quality CASSETTES £4.90 for 10

Plus the opportunity to join the fastest moving club of personal computer users enabling you to get the most our of your computer. You can OBTAIN and EXCHANGE programs and other software – many now available.

The Powerful Z80 Microprocessor
Professional Keyboard
1 Kbyte Monitor in EPROM
2 Kbyte RAM (expandable)
Audio Cassette interface
Plugs into your domestic TV
Easy construction from straightforward instructions
– no drilling or special tools
– Just neat soldering required.



Only **£197.50** + 8% VAT (includes p & p + insurance)

Manuals seperately	3.50	Monitor quality improved	
Z80 programming Manual	6.90	TV Modulator	2.50
Z80 Technical Manual	3.40	Power supply suitable for	
PIO Technical Manual	3.40	NASCOM	19.90

(All prices add 8% VAT)

COMPUTER COMPONENTS

A selection of our Computer components

CPU AND SUPPORT DEVICES

Z80	cpu	14.90
Z80A	cpu	20.50
Z80	PIO	9.90
Z80	CTC	9.90
8080A	cpu	7.95
8212	8 Bit I/O Port	3.50
8214	Priority Interrupt Control	9.00
8216	Bi-Directional Bus Driver	3.75
8224	Clock Generator & Driver (2MH$_z$)	3.50
8224-4	Clock Generator & Driver (4MH$_z$)	9.95
8226	Inverting Bi-Directional Bus Driver	3.95
8228	System Controller & Bus Driver	7.95
8238	System Controller & Bus Driver	7.50
8251	Programmable Communication Int.	9.95
8253	Programmable Interval Timer	18.95
8255	Programmable Peripheral Int.	12.00
8257	Prog. Direct Mem. Access Cont.	17.95
8259	Programmable Interrupt Cont.	17.95
6800	cpu	11.90

MEMORIES

2102	1K	1.20
21L02	1K	1.40
2101	1K	2.95
2111	1K	3.95
2112	1K	2.95
2114	4K	9.95
4116	16K	19.90
1702A	256B	5.00
2708	1K	10.90
2708	(NASBUG)	13.90
2716	(5V, INTEL)	29.00

OTHER USEFUL DEVICES

6576	char gen	10.90
6402	UART	9.90
1771	Floppy disk controller	49.00

Full selection of US and British Magazines and books for the Computer Hobbyist.

ETI 8080 ONE BOARD COMPUTER

COMPONENTS FOR THE ELECTRONICS TODAY INTERNATIONAL MAGAZINE COMPUTER PROJECT.
GREAT VALUE
SEE ETI NOVEMBER 1978!
SEND SAE FOR PRICE LIST OF PARTS AND KIT.

All prices include VAT except where shown. Orders over £5 post and packing free otherwise add 20p. Please make cheques and postal orders payable to COMP, or phone your order quoting BARCLAYCARD or ACCESS number.

BARCLAYCARD

Buy it with Access

COMP COMPUTER COMPONENTS

14 STATION ROAD · NEW BARNET · HERTFORDSHIRE · TEL: 01-441 2922
CLOSE TO NEW BARNET BR STATION MOORGATE LINE 01-449 6596
OPEN – 10am to 7pm – Monday to Saturday

ALIGNING THE DECIMAL POINT

Sidney A. Leleux

Many programs, especially those for accountancy purposes, require columns of figures in the output to have the decimal points in line. I have found the following coding to serve the purpose well with a minimum of effort. The hardest part is to make sure that the brackets match!

Variables
- N the number to be printed
- T the number of TAB spaces – would probably be a constant in a given PRINT statement.

```
100 IF ABS(N)<1 THEN 130
110 PRINT TAB(T-INT(LOG(ABS(N))/LOG(10)));N
120 GOTO 140
130 PRINT TAB(T);N
140 etc.
```

The coding works by calculating the characteristic of the normal logarithm of the number – and thus the number of places left of the point. This quantity is then subtracted from the basic print position.

The ABS function is necessary as you cannot calculate the logarithm of a negative number. The computer's LOG function calculates logarithms "to the base e" (e = 2.71 . . . . which is mathematically a very useful number!). Division by LOG(10) converts the logarithm to base 10. INT extracts the characteristic.

Line 100 takes care of numbers numerically less than 1.

Their negative characteristics move the point to the right of the coumn if they enter line 110! Line 100 also deals with the case of N = O, which would cause an error when an attempt was made to evaluate LOG(0).

Since positive numbers are printed with a leading space (at least in the versions of BASIC which I have met) there is no need to allow for the space taken by the minus sign required by negative numbers.

This coding is suitable for numbers printed in normal format. It fails once "E format" numbers are encountered, but as E-format is only used outside the range $10^6 > N \geq 0.01$ (or similar) ordinary numbers are adequately served.

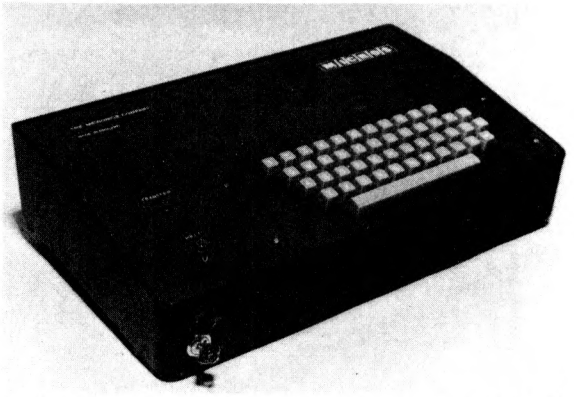
ANNOUNCING THE MICROS

JOIN THE MICRO REVOLUTION

Access We welcome Access

master charge THE INTERBANK CARD

EUROCARD



£399 for a Z80 based microcomputer, built and tested

Designed for educational establishments, personal computing and small business users

* Includes 1K monitor Eprom, 47 key solid state keyboard, video, TV, cassette and teletypewriter interfaces,serial i/o, 2 parallel i/o ports, 2K bytes RAM, power supplies and instrument housing.
* Connect to domestic TV or video monitor to complete the system
* 48 x 16 character video matrix
* 47 key contactless ASC11 keyboard
* Hard copy on teletypewriter
* 2 TTL compatible parallel i/o ports
* RS232 serial i/o port
* Load and dump programmes on unmodified cassette recorder
* Up to 16K byte mixed RAM and Eprom in table top housing
* Expandable up to 64K bytes
* Security locked power switch
* British designed and built
* Available in kit form for £360
* Credit terms available

THE MICRONICS COMPANY

1, STATION ROAD TWICKENHAM MIDDLESEX

01-892 7044

PART OF THE MICRO REVOLUTION

Prices exclusive of VAT and carriage

Getting It Together

Build your own assembler — listing (continued)

Mike Banahan

```
0635 CDB10F7DD604F2D403          CALL EVALUA#A<-L#SUB#10#J P,OTH,BX
063E 7DCB27CB27CB27              A<-L#SLA A#SLA A#SLA A
0645 47CD3601                    B<-A#CALL GETATM
0649 21080DCDD30E2000            HL<-#SRTAB#CALL SEARCH#BM NZ,SETR.1
0651 3ECBCDFD11                  A<-#313#CALL OUTINS
0656 3A2406B0B6CB7DA003          A<-SETF#OR B#OR L#J OTH,OX
065E CD3108              SETR.1: CALL HLIXIY
0661 3ECBCDFD11                  A<-#313#CALL OUTINS
0666 7AB779C4FD11                A<-D#TSTA#A<-C#CALL NZ,OUTINS
066C 3A2406B0F60AC3DAB3          A<-SETF#OR B#OR#6#J OTH,OX
0675 04                  SRL:    INC B
0676 04                          INC B#( NON-DOCUMENTED OPCODE
0677 04                  SRA:    INC B
0678 04                  SLA:    INC B
0679 04                  RR:     INC B
067A 04                  RL:     INC B
067B 04                  RRC:    INC B
067C CB20CB20CB20        RLC:    SLA B#SLA B#SLA B
06E2 21080DCDD30E200B            HL<-#SRTAB#CALL SEARCH#BM NZ,RLC.1
068A 3ECBCDFD11                  A<-#313#CALL OUTINS
068F AFB0B5C3DA03                CLA#OR B#OR L#J OTH,OX
0695                     RLC.1:
0695 218808CDD30E                HL<-#OTHTAB#CALL SEARCH
069B CD3601                      CALL Z,GETATM#  (DEALS WITH RLC(HL)
069E                                             (RLC (HL) O.K. ANYWAY
069E                                             (TOUGH ABOUT RLC RLC(IX))
069E CD3108                      CALL HLIXIY
06A1 3ECBCDFD11                  A<-#313#CALL OUTINS
06A6 7AB779C4FD11                A<-D#TSTA#A<-C#CALL NZ,OUTINS
06AC 3E06B0C3DA03                A<-#6#OR B#J OTH,OX
06B2 21080DCDD30E280C    IN:     HL<-#SRTAB#CALL SEARCH#BM Z,IN.2
06BA 3AAC15FE46C2D403            A<-ATBLCK#CMP#'F#J NZ,OTH,BX
06C2 2E062630                    L<-#6#HC-#60#    (SOME DAM' FUNNY INSTRUCTIONS ABOUT
06C6 CDB703C2D403        IN.2:   CALL COMCHK#J NZ,OTH,BX
06CC 3E07BD2017                  A<-#7#CMP L#BR NZ,IN.1#(          IS IT IN A,X?
06D1 CD360108FE28281A            CALL GETATM#EX AF#CMP #'(#BR Z,IN.3
06D9 CDB10FC4F512                CALL EVALUA#CALL NZ,TRERM
06DF 3E0BCDFD11                  A<-#333#CALL OUTINS
06E4 7DC3DA03                    A<-L#J OTH,OX
06E8 CD360108            IN.1:   CALL GETATM#EX AF
06EC FE28C2D403                  CMP#'(#J NZ,OTH,BX
06F1 CD3601              IN.3:   CALL GETATM
06F4 3AAC15FE43C2D403            A<-ATBLCK#CMP#'C#J NZ,OTH,BX#(  IT'S IN R,(C....
06FC 3EEDCDFD11                  A<-#355#CALL OUTINS
0701 3E40B4C3DA03                A<-#100#OR H#J OTH,OX
0707 3AB811FE28C824      OUT:    A<-TERMIT#CMP#'(#BR Z,OUT.1
070E CDB703C2D403                CALL COMCHK#J NZ,OTH,BX
0714 CDB10FC4F512                CALL EVALUA#CALL NZ,TRERM
071A 45CD3601210A0D              B<-L#CALL GETATM#HL<-#SRTAB
0721 CDD30EFE07C2D403            CALL SEARCH#CMP#7#J NZ,OTH,BX
0729 3ED3CDFD11                  A<-#323#CALL OUTINS
072E 78C3DA03                    A<-B#J OTH,OX
0732 CD36013AAC15        OUT.1:  CALL GETATM#A<-ATBLCK
0738 FE43C2D403                  CMP#'C#J NZ,OTH,BX
073D 3AB811FE29C2D403            A<-TERMIT#CMP#')#J NZ,OTH,BX
0745 CD3601CDB703C2D403          CALL GETATM#CALL COMCHK#J NZ,OTH,BX
074E CD360121040DCDD70E          CALL GETATM#HL<-#SRTAB#CALL SEARCH
```

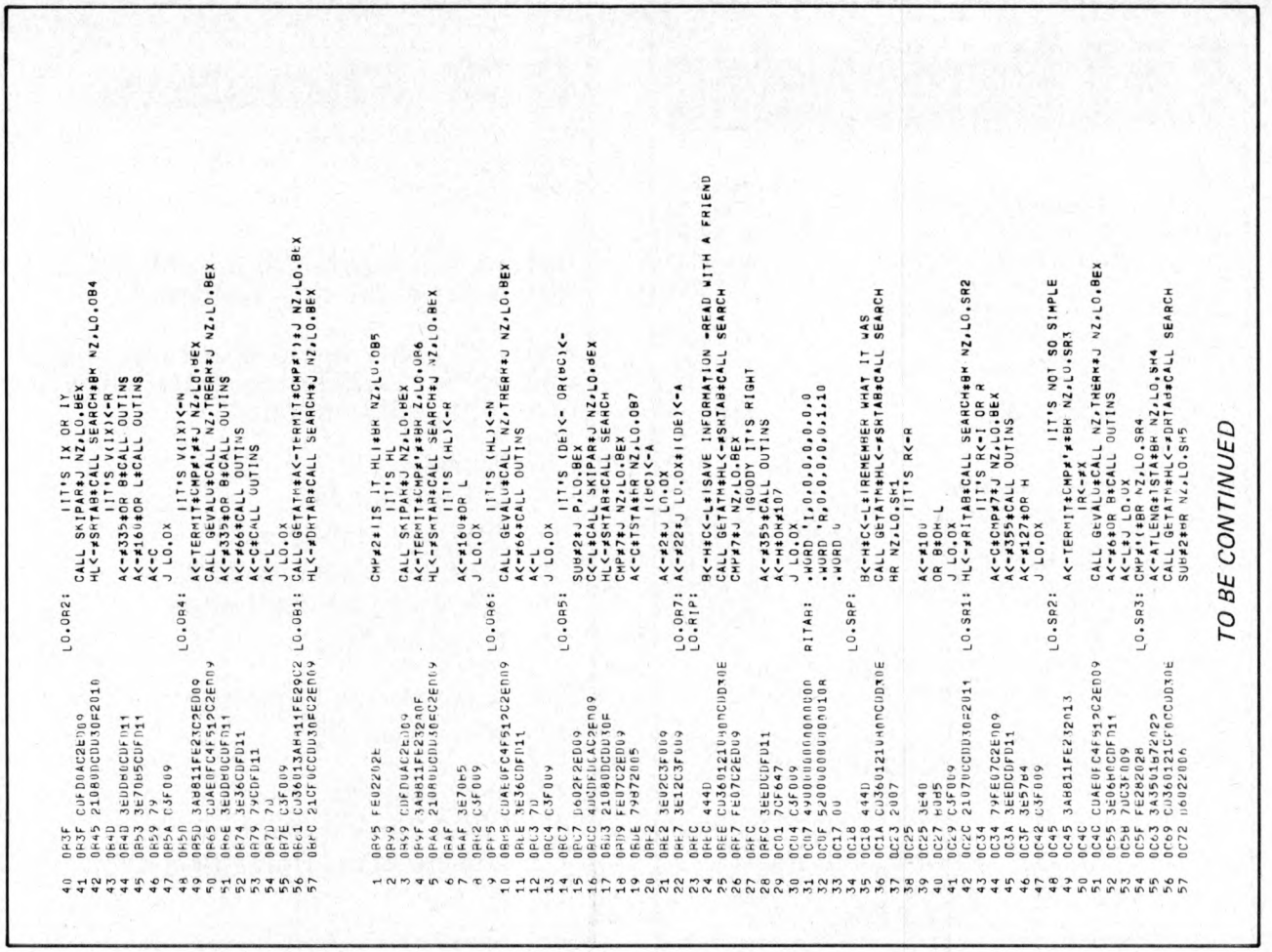
```
0757 C2D403                      J NZ,OTH,BX
075A 3EEDCDFD11                  A<-#355#CALL OUTINS
075F 3E41B4C3DA03                A<-#101#OR H#J OTH,OX
0765 04                  DECL:   INC B
0766 3E04B047            INCL:   A<-#4#OR B#B<-A
076A 21080DCDD30E2005            HL<-#SRTAB#CALL SEARCH#BM NZ,INCL.1
0772 AFB4B0                      CLA#OR H#OR B
0775 1834                        BR INC.OX
0777 21CF0CCDD30E2811    INCL.1: HL<-#DRTAB#CALL SEARCH#BM Z,INCL.2
077F CD3108                      CALL HLIXIY
0782 3E30B0CDFD11                A<-#60#OR B#CALL OUTINS
0788 7AB7CADDD3                  A<-D#TSTA#J Z,OTH,GX
078D 79                          A<-C
078E 181B                        BR INC.OX
0790 E6082808            INCL.2: AND#10#BR Z,INCL.3
0794 3EDDB4CDFD11                A<-#335#OR H#CALL OUTINS
079A 2620                        H<-#40
079C 70FE07CAD403        INCL.3: A<-L#CMP#7#J Z,OTH,BX
07A2 78878787E60F                A<-B#ADD A#ADD A#ADD A#AND#10
07A8 F603B4                      OR#3#OR H
07AB CDFD11CDB703C2DD03  INC.OX: CALL OUTINS#CALL COMCHK#J NZ,OTH,GX
07B4 CD3601                      CALL GETATM
07B7 78D600FE042AA8              A<-B#B<-#0#CMP#4#BR Z,INCL
07BE 18A5                        BR DECL
07C0 3AB811FE282A30      EX:     A<-TERMIT#CMP#'(#BR Z,EX.2
07C7 21CF0CCDD30E                HL<-#DRTAB#CALL SEARCH
07CD C2D403                      J NZ,OTH,BX
07D0 FE072005                    CMP#7#BR NZ,EX.1
07D4 3E08C3DAA3                  A<-#10#J OTH,OX#(          IT'S EX AF
07D9 FE01C2D403CDB703C2  EX.1:   CMP#1#J NZ,OTH,BX#CALL COMCHK#J NZ,OTH,BX
07E4 CD360121CF0CCDD30E          CALL GETATM#HL<-#DRTAB#CALL SEARCH
07ED FE02C2D403                  CMP#2#J NZ,OTH,BX#(      EX DE,HL?
07F2 3EEB                        A<-#353
07F4 C3DA03                      J OTH,OX
07F7 CD36013AB811FE29    EX.2:   CALL GETATM#A<-TERMIT#CMP#')
07FF C2D403                      J NZ,OTH,BX
0802 21CF0CCDD30E                HL<-#DRTAB#CALL SEARCH
0808 FE03C2D403                  CMP#3#J NZ,OTH,BX
080D                             (WONDERFUL. IT'S EX (SP)
080D CD3601CDB703C2D403          CALL GETATM#CALL COMCHK#J NZ,OTH,BX
0816 CD3601                      CALL GETATM
0819 21CF0CCDD30EC2D403          HL<-#DRTAB#CALL SEARCH#J NZ,OTH,BX
0822 FE022806                    CMP#2#BR Z,EX.3
0826 3EDDB4CDFD11                A<-#335#OR H#CALL OUTINS#(       EX SP,(IX OR IY)
082C 3EE3C3DA03          EX.3:   A<-#343#J OTH,OX
0831
0831                     (       HLIXIY IS USED ALL OVER THE PLACE,
0831                     (       IT ALLOWS ONE TO DEAL WITH (HL), OFFSET(IX OR IY)
0831                     (       AS IF THEY WERE ALL (HL)
0831                     (       IF OFFSET IS PRESENT, D EXITS NON ZERO
0831                     (       AND C CONTAINS THE OFFSET.
0831
0831                     HLIXIY:
0831 3A3501B7572809              A<-ATLENG#TSTA#DC-A#BR Z,HL.1
0838 CDB10FC4F5122033            CALL EVALUA#CALL NZ,TRERM#BR NZ,HLBRET
0840 4D                          C<-L
0841 3AB811FE282028      HL.1:   A<-TERMIT#CMP#'(#BR NZ,HLBRET
```

```
0A48 CD3601             CALL GETATM
0A4B 3AB811FE2820       A<-TERMIT#CMP#'(')#BR NZ,HL#RET
0A52 21CF0CD5CDD30ED1   HL<-#DHTAB#PUSH DE#CALL SEARCH#POP DE
0A5A FE02C006   HL.2:   CMP#2#BR NZ,HL.3
0A5E 7AB72011           A<-D#TSTA#BR NZ,HL#RET
0A62 1E0E               BR HLGRET#      ICAN'T INDEX HL
0A64 E6082808   HL.3:   AND#10#BR Z,HLHRET
0A68 7AB728D7           A<-D#TSTA#BR Z,HLHRET#  ICAN'T HAVE INC(IX) DUCKY
0A6C 3EDDB4CDFD11       A<-#335#OR H#CALL OUTIN$
0A72 C9         HLGRET: RET
0A73 E1C3D463   HLHRET: POP HL#I UNWIND STACK!#J OTH.BX
0A77 4E5A000000000000   CONTAB: .WORD 'N,'Z,0,0,0,0,0,0
0A7F 5A0000000000010A           .WORD 'Z,0,0,0,0,0,1,10
0A87 4E4300000000021A           .WORD 'N,'C,0,0,0,0,2,20
0A8F 430000000000031A           .WORD 'C,0,0,0,0,0,3,30
0A97 504F00000000042A           .WORD 'P,'O,0,0,0,0,4,40
0A9F 5045000000000052A          .WORD 'P,'E,0,0,0,0,5,50
0AA7 500000000000063A           .WORD 'P,0,0,0,0,0,6,60
0AAF 4D000000000073A            .WORD 'M,0,0,0,0,0,7,70
0AB7 00                         .WORD 0
0AB8 525354000000E20A   OTHTAB: .WORD 'R,'S,'T,0,0,0*.ADDR RST
0AC0 43414C4C0000F40A           .WORD 'C,'A,'L,'L,0,0*.ADDR CALL
0AC8 44424E5A0000F20A           .WORD 'D,'B,'N,'Z,0,0*.ADDR DBNZ
0AD0 4252000000005F0A           .WORD 'B,'R,0,0,0,0*.ADDR BR
0AD8 4A000000000C8C0A           .WORD 'J,0,0,0,0,0*.ADDR J
0AE0 5245540000022204           .WORD 'R,'E,'T,0,0,0*.ADDR RETCC
0AE8 5055534B0000F204           .WORD 'P,'U,'S,'H,0,0*.ADDR PUSH
0AF0 504F5000000F6004           .WORD 'P,'O,'P,0,0,0*.ADDR POP
0AF8 4558000000000C007          .WORD 'E,'X,0,0,0,0*.ADDR EX
0B00 434D5000000001D05          .WORD 'C,'M,'P,0,0,0*.ADDR CMP
0B08 4F52000000001E05           .WORD 'O,'R,0,0,0,0*.ADDR OR
0B10 5848520000001F05           .WORD 'X,'O,'R,0,0,0*.ADDR XOR
0B18 414E4400000020C5           .WORD 'A,'N,'D,0,0,0*.ADDR AND
0B20 5342430000022105           .WORD 'S,'B,'C,0,0,0*.ADDR SBC
0B28 5355420000022205           .WORD 'S,'U,'B,0,0,0*.ADDR SUBL
0B30 4144430000023205           .WORD 'A,'D,'C,0,0,0*.ADDR ADC
0B38 4144440000024205           .WORD 'A,'D,'D,0,0,0*.ADDR ADDL
0B40 494E43000000660?           .WORD 'I,'N,'C,0,0,0*.ADDR INCL
0B48 444543000000650?           .WORD 'D,'E,'C,0,0,0*.ADDR DECL
0B50 494E0000000008206          .WORD 'I,'N,0,0,0,0*.ADDR IN
0B58 4F5554000000070?           .WORD 'O,'U,'T,0,0,0*.ADDR OUT
0B60 5445535400000E505          .WORD 'T,'E,'S,'T,0,0*.ADDR TEST
0B68 5345540000025006           .WORD 'S,'E,'T,0,0,0*.ADDR SET
0B70 53455424144D0B04           .WORD 'S,'E,'T,'R,'A,'D#.ADDR SETRAD
0B78 5245534554000290A          .WORD 'R,'E,'S,'E,'T,0*.ADDR RESET
0B80 53524C0000007506           .WORD 'S,'R,'L,0,0,0*.ADDR SRL
0B88 5352410000007706           .WORD 'S,'R,'A,0,0,0*.ADDR SRA
0B90 534C410000007806           .WORD 'S,'L,'A,0,0,0*.ADDR SLA
0B98 5252000000007906           .WORD 'R,'R,0,0,0,0*.ADDR RR
0BA0 524C000000007A06           .WORD 'R,'L,0,0,0,0*.ADDR RL
0BA8 5252430000007B06           .WORD 'R,'R,'C,0,0,0*.ADDR RRC
0BB0 524C430000007C06           .WORD 'R,'L,'C,0,0,0*.ADDR RLC
0BB8 00                         .WORD 0
0BB9
0BB9                    !       MESSY BIT DEALING WITH ALL THE LOAD
0BB9                    !       INSTRUCTIONS. IT'S ONLY GRACE IS THAT MUCH
0BB9                    !       OF IT SEEMS TO WORK.
```

```
0BB9
0BB9                    LOAD:
0BB9 F5C5D5E5                   PUSH AF,BC,DE,HL
0BBD 3AB811FE28CA0B0B           A<-TERMIT#CMP#'(#J Z,LO.OBR
0BC5 3A3501B7CAA90A             A<-ATLENG#!STA#J Z,LO.R2M
0BCC 2108DCDD30FCA1A0C          HL<-#SRTAB#CALL SEARCH#J Z,LO.SRP
0BD5 2107DCDD30FCAEC0B          HL<-#RITAB#CALL SEARCH#J Z,LO.RIP
0BDE 21CF0CDD30FCA0D0A          HL<-#DHTAB#CALL SEARCH#J Z,LO.DRP
0BE7 CD810FC3A80A               CALL EVALUA#J LO.R2M
0BED CDED12             LO.BEX: CALL SYNTEX
0BF0 CDFD11             LO.OX:  CALL OUTINS
0BF3 F1D1C1F1C9         LO.GEX: POP HL,DE,BC,AF#RET
0BF8 F57DCDFD11         OUTHL:  PUSH AF#A<-L#CALL OUTINS
0BFD 7CCDFD11                   A<-H#CALL OUTINS
0A01 F1C9                       POP AF#RET
0A03 C5E5               OUTBC:  PUSH BC,HL
0A05 6069CDFB09                 H<-B#L<-C#CALL OUTHL
0A0A E1C1C9                     POP HL,BC#RET
0A0D                    LO.DRP:
```

```
0A0D 444D                       H<-H#C<-L
0A0F                                    !REMEMBER WHICH REG PAIR
0A0F CD3601                     CALL GETATM
0A12 3AB811FE23                 A<-TERMIT#CMP#'#
0A17 202E                       BR NZ,LO.DR1
0A19 CDAE0F                     CALL GEVALU
0A1C 79E6B2B810                 A<-C#AND#370#BR Z,LO.DR2
0A21                                    !IT'S IX OR IY
0A21 3EDDB4CDFD11               A<-#335#OR B#CALL OUTINS
0A27 3E21CDFD11                 A<-#041#CALL OUTINS
0A2C CDF809                     CALL OUTHL
0A2F 18C2                       BR LO.GEX
0A31 79FE07CAED09       LO.DR2: A<-C#CMP#7#J Z,LO.HEX#!IF IT'S AF
0A37 78?CCDFD11                 A<-B#!INC A#CALL OUTINS
0A3C CDF809                     CALL OUTHL
0A3F C3F309                     J LO.GEX
0A42 3EF9C3F009         LO.DR6: A<-#371#J LO.OX#!SP<-HL
0A47 79FE03?019         LO.DR1: A<-C#CMP#3#BR NZ,LO.DR4
0A4C                                    !IT'S SP<-
0A4C 21CF0C                     HL<-#DHTAB
0A4F CDD30FFE022AEC             CALL SEARCH#CMP#2#BR Z,LO.DR6
0A56 E6082808                   AND#10#BR Z,LO.DR4#!RELIES ON L BEING ZERO
0A5A                                    !IF SEARCH FAILS
0A5A                                    !IT'S SP<-IX OR IY
0A5A 3EDDB4CDFD11               A<-#335#OR H#CALL OUTINS
0A60 3EF9                       A<-#371
0A62 C3F009                     J LO.OX
0A65 CD810F             LO.DR4: CALL EVALUA
0A68 79FE022815         LO.DR8: A<-C#CMP#2#BR Z,LO.DR9#!IT'S HL<-
0A6D E6F8201C                   AND #370#BR NZ,LODR10#!IX OR IY
0A71 3EEDCDFD11                 A<-#355#CALL OUTINS
0A76 78                         A<-B
0A77 F64BCDFD11                 OR#113#CALL OUTINS
0A7C CDF809C3F309               CALL OUTHL#J LO.GEX
0A82 3E2ACDFD11         LO.DR9: A<-#52#CALL OUTINS
0A87 CDF809C3F309               CALL OUTHL#J LO.GEX
0A8D                    LODR10:         !IT'S IX OR IY
0A8D 3EDDB4CDFD11               A<-#335#OR B#CALL OUTINS
```

```
0A93 3E2ACDFD11                 A<-#52#CALL OUTINS
0A98 CDF809C3F309               CALL OUTHL#J LO.GEX
0A9E 3E22CDFD116069     LO.RM1: A<-#42#CALL OUTINS#H<-B#L<-C
0AA5 CDF809C3F309               CALL OUTHL#J LO.GEX
0AAB                    LO.R2M:
0AAB 444D                       B<-H#C<-L
0AAD CD3601                     CALL GETATM
0AB0 21CF0CDD30FE291A           HL<-#DHTAB#CALL SEARCH#BR Z,LO.RM4
0AB8 2108DCDD30F                HL<-#SHTAB#CALL SEARCH
0ABE FE07C2ED09                 CMP#7#J NZ,LO.BEX
0AC3                                    !CONST<-A
0AC3 3E32CDFD11                 A<-#62#CALL OUTINS
0AC8 CD030AC3F309               CALL OUTBC#J LO.GEX
0ACE FE02280C           LO.RM4: CMP#2#HR Z,LO.RM1
0AD2 FE07CAED09                 CMP#7#J Z,LO.BEX#!CAN'T DO CONST<-AF
0AD7 E6F82811                   AND#370#BR Z,LO.RM3
0ADB                                    !IT'S IY OR IX
0ADB 3EDDB4CDFD11               A<-#335#OR H#CALL OUTINS
0AE1 3E22CDFD11                 A<-#042#CALL OUTINS
0AE6 CD030A                     CALL OUTBC
0AE9 C3F309                     J LO.GEX
0AEC 3EEDCDFD11         LO.RM3: A<-#355#CALL OUTINS
0AF1 7C                         A<-H
0AF2 F643CDFD11                 OR#103#CALL OUTINS
0AF7 CD030A                     CALL OUTBC
0AFA C3F309                     J LO.GEX
0AFD CD36013ABA11       SKIPAR: CALL GETATM#A<-TERMIT
0B03 FE5FC0                     CMP#'<-#RET NZ
0B06 CD3601AF                   CALL GETATM#CLA
0B0A C9                         RET
0B0B                    LO.OBR:
0B0B 3A3501B7CA810B             A<-ATLENG#!STA#J Z,LO.OB1#!LOOK FOR VALUE
0B12 2108DCDD30F                HL<-#OTHTAB#CALL SEARCH
0B18 CA8D03CAF309               CALL Z,OTHERS#J Z,LO.GEX
0B1E CD810FC4F512C2EB09         CALL EVALUA#CALL NZ,THERM#J NZ,LO.BEX
0B27 4D                         C<-L
0B28 CD36013ABA11FE29C2         CALL GETATM#A<-TERMIT#CMP#')#J NZ,LO.BEX
0B33 21CF0CDD30F                HL<-#DHTAB#CALL SEARCH
0B39 E608CAED0944               AND#10#J Z,LO.BEX#B<-H
```



```
                        IIT'S IX OR IY
LO.0A2:                 CALL SKIPAR#J NZ,LO.BEX
0F3F CDFD0AC2E7D0       HL<-#SNTAB#CALL SEARCH#BH NZ,LO.0B4
0F45 2108D0CDD30F2010           IIT'S V(IX)<-R
0F4D 3EDDB6CDFD11               A<-#33>#OR B#CALL OUTINS
0F53 3E70B5CDFD11               A<-#16U#OR L#CALL OUTINS
0F59 79                         A<-C
0F5A C3F009             J LO.0X
                        IIT'S V(IX)<-N
0F5D 3A8811FE23CAED09   A<-TERMIT#CMP#'#'J NZ,LO.0EX
0F65 CDAE0FC4F51AC2E009 CALL GETVAL#CALL NZ,TREAM#J NZ,LO.0EX
0F6E 3EDDB6CDFD11       A<-#33>#OR B#CALL OUTINS
0F74 3E36CDFD11         A<-#66#CALL OUTINS
0F79 79CDFD11           A<-C#CALL OUTINS
0F7D C3F009             A<-L
0F80 C3F009             J LO.0X
0F83 CD361213AH11FE29C2 LO.0B1: CALL GETATH#AC<-TERMIT#CMP#')'#J NZ,LO.0EX
0F8C 21CF0CCDD30FC2E009         HL<-#DATA#CALL SEARCH#J NZ,LO.0EX


0F95 FE022B2E           CMP#2#IS IT HL1#BH NZ,LO.0B5
0F99                            IIT'S HL
0F99 CDFD0AC2E3D9       CALL SKIPAR#J NZ,LO.0EX
0F9F 3A8811FE23220F     AC<-TERMIT#CMP#'#'BH Z,LO.0F6
0FA6 2108D0CDD30FC2E009 HL<-#NTAB#CALL SEARCH#J NZ,LO.0EX
0FAF 3E70B5             A<-#16U#OR L
0FB2 C3F009             J LO.0X
                        IIT'S (HL)<-N
0FB5 CDAE0FC4F51AC2E009 CALL GETVAL#CALL NZ,TREAM#J NZ,LO.0EX
0FBE 3E36CDFD11         A<-#66#CALL OUTINS
0FC3 79                 A<-L
0FC4 C3F009             J LO.0X
                        IIT'S (DE)<- OR(HL)<-
0FC7 D602F2ED09         SUB#2#J P,LO.0EX
0FCC 40CDF0CAC3E009     CC<-L#CALL SKIPAR#J NZ,LO.0EX
0FD3 2108D0CDD30F       HL<-#NTAB#CALL SEARCH
0FD9 FE07C2ED09         CMP#7#J NZ,LO.0EX
0FDE 79B72005           AC<-C#TSTA#R NZ,LO.0B7
0FE2                            (HL)<-A
0FE2 3EU2C3F009         A<-#2#J LO.0X
0FE7 3E12C3F009         A<-#22#J LO.0X#(DE)<-A
0FEC 4449               BC<H#CC<-L#ISAVE INFORMATION -READ WITH A FRIEND
0FEE CD361213AH0CCDD30E CALL GETATH#HL<-#SNTAB#CALL SEARCH
0FF7 FE07C2ED09         CMP#7#J NZ,LO.0EX
0FFC                            IGUODY IT'S RIGHT
0FFC 3EEDCDFD11         A<-#35>#CALL OUTINS
1001 7DF647             A<-H#OR#107
1004 C3F009             J LO.0X
1007 4900000000000000   .WORD 'I,0,0,0,0,0,0,0
100F 5200000000000010   .WORD 'R,0,0,0,0,0,1,10
1017 00                 .WORD 0
1018                            BC<H#CC<-L#IREMEMBER WHAT IT WAS
101A CD361213AH0CCDD30E CALL GETATH#HL<-#SNTAB#CALL SEARCH
1023 2007               HR NZ,LO.SH1
1025                            IIT'S R<-R
1025 3E40               A<-#10U
1027 B0D5               OR B#OH L
1029 C3F009             J LO.0X
102C 21D7DCDD30F2011    LO.SH1: HL<-#RITAR#CALL SEARCH#BH NZ,LO.SR2
1034                            IIT'S R<-I OR R
1034 79FE0702E009       AC<-C#CMP#7#J NZ,LO.0EX
103A 3EEDCDFD11         A<-#35>#CALL OUTINS
103F 3E57B4             A<-#127#OR H
1042 C3F009             J LO.0X
1045                            IIT'S NOT SO SIMPLE
1045 3A8811FE212013     LO.SR2: AC<-TERMIT#CMP#'#'BH NZ,LO.SR3
                                IR<-#X
104C CDAE0FC4F51AC2E009 CALL GETVAL#CALL NZ,TREAM#J NZ,LO.0EX
1055 3E06B6CDFD11       A<-#6#OR B#CALL OUTINS
105B 79C3F009           A<-L#J LO.0X
105F FE2B2028           CMP#'(#BR NZ,LO.SR4
1063 3A3501B720D2       AC<-ATLENG#ISTA#BH NZ,LO.SH4
1069 CD361213CF0CCDD30E CALL GETATH#HL<-#DATA#CALL SEARCH
1072 1602C2D06          SUB#2#R NZ,LO.SH5
```

TO BE CONTINUED

EQUINOX 300

A powerful multi-user
multi-tasking
multi-language

16-bit microcomputer time-sharing system

supporting
* BASIC
* LISP
* PASCAL
* Floppy Discs
* Hard discs

including a powerful Text Formatter, Assembly Language Development System and disc based Sort utilities.

Priced from under £5,000

Write or phone for further information

EQUINOX COMPUTER SYSTEMS LTD.
32-35 Featherstone Street,
London, EC1Y 8QX
(01-253 3781/9837)

MOTOROLA 6800 COMPUTERS

Single board 6800 17 command 1K Mikbug compatible Monitor. 1K user RAM, 1K Crystal Controlled VDU. CUTS, fully buffered, room for extra 3 x 2708 PROMS **£175.00**
Single board 2. Similar to above but with QWERTY keyboard **£205.00**
NEW Mini 6800. Two p.c.b. kit. 1K monitor. 1K VDU, RAM, CUTS, QWERTY keyboard . . . **£145.00**
VDU kit. 1K Crystal controlled, memory mapped, parallel input video output. Should suit most processors. **£60.00**

All prices exclude VAT and post. Please send SAE for leaflets. Mail Order only.

HEWART MICROELECTRONICS

95 Blakelow Road
Macclesfield, Cheshire

V.&T.Electronics

KITS

Nascom I, Microcomputer Kit		£197.50
Jade Z80 CPU Kit 2MHZ		£ 95.00
Jade 8K Static Memory 250NS		£119.95
Jade 32K Static Memory 250NS		£595.00
Jade 8K Eprom Board With 8K of 2708 + Programmer; Kit		£118.50
13 Slot S100 Motherboard		£ 48.50
KB756 Keyboard Kit		£ 45.00
Mini Disk Drive (ready built)		£200.00
I:Cs		
2708 1K x 8 Eprom 450NS		£ 4.50
21L02 1K x 1 Static 450NS	8OFF	£ 7.90
21L02 1K x 1 Static 250NS	8OFF	£ 9.50
2107 4K x 1 Dynamic 270NS	8OFF	£ 25.00
4116 16K x 1 Dynamic 200NS	8OFF	£100.00
Z80 CPU 2MHZ		£ 13.00

Full Range of TTL & CMOS Stocked
Please Write or Phone for Quote and Price List.

Retail Store Opening Beginning of November 1978
Please add 40p postage and packing them 8% V.A.T.

V. & T. ELECTRONICS
82 CHESTER ROAD,
LONDON, N.19 5BZ
01-263 2643

CCS Microhire

MICROCOMPUTER RENTAL SPECIALISTS

Before you buy a micro, why not hire it for a day or a weekend?

From £2 a day we hire out a range of micros for evaluation/experience or program development.

* Apple II
* Commodore Pet
* Nascom I or Micros
* Research Machines 380Z
* SWTC 6800 or MSI 6800
* Tandy TRS 80

Protect your future investment.
Try out a system now.

For details write to
CCS Microhire, Freepost,
Letchworth, Herts SG6 4YA

JUST CONNECT AN AMPLIFIER OR CASSETTE RECORDER TO THE BACK OF YOUR PET AND ADD SOUND EFFECTS AND TUNES TO YOUR PROGRAMMES. NO INTERFACE NECESSARY.

THE FOLLOWING PROGRAMMES ARE AVAILABLE ON CASSETTE:--

1. **MOON LANDING** **£6 (With Sound Effects)**
2. **TUNES I** **£3 (Pop + Folk)**
3. **TUNES II** **£3 (Traditional)**

Cheque or P.O. to:— J. H. Macleod, 11 St. Martins Drive, Eynsford, Kent, DA4 0EY.

STARSHIP SIMULATION. By Roger Garrett £5.60. A FAST INTRODUCTION TO COMPUTING £3.45, both including post, or S.A.E. for list of computing books.
Industrial Training Press, dept. PCW, 3 Ringwood Way, Winchmore Hill, London N21 2RA.

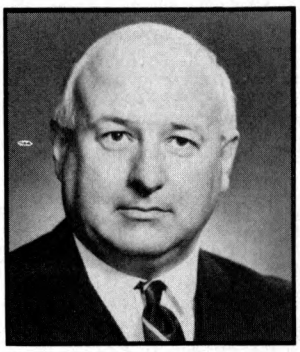
MICRO & FLOPPY DISC
FOR HIRE
FROM £3.50 per hour
Programs & Support
061-224 1581

PHREAKSTEIN



"MAY I INTRODUCE"...

Using a Small Computer



P.M. Steele
Director,
Peter Steele and Partners Ltl
1624 High Street, Knowles,
Solihull B93 0JU

This article describes a search for a better solution to a problem common to many professional organisations, technical publications and smaller companies. These bodies need to keep up-to-date records of subscribers, members or customers who need servicing periodically and/or invoicing. The problem falls into two parts from a systems analysis viewpoint – the first is to develop an up-to-date master file which is best kept in a format that can be visually checked and organised i.e. it needs to be written on file cards rather than exist as a computer tape or disk as the primary record. This master record needs usually to be updated with additions, deletions, or changes of name of address at weekly or monthly intervals.

The second part of the problem involves the translation of the data held in the master file into various physical formats, such as addressed labels or invoices.

Staff Training Problems

Many professional publications or learned society offices, for example, have an editorial and office staff highly qualified to produce copy about medicine, textile machinery, etc, but who are not experienced office workers who will operate a conventional stencil addressing machine system with the skill and attention to detail necessary to prevent the records getting in a serious muddle.

Many of these publications and professional organisations distribute material on a world-wide basis. In this article a technical magazine is used as an example. It sends copies to China, Japan, Russia, Balkan countries, European countries and many other places where the addresses are not easily meaningful to the average English typist.

Potential Loss

Since each magazine costs £1.00, mis-addressed labels represent a serious direct cost to the publisher, plus a great deal of correspondence spread over many months dealing with requests for non-delivered copies.

In many cases the journals go to places served by very rudimentary postal services, so that the labels have to be very clear and have a positive bond to the journal packaging. To achieve this high quality requires technical co-operation between the label-maker, label-user and the packer of the journals. With plastic outer envelopes, it was necessary to seek the co-operation of the manufacturer of the resins on the label in order to provide a bond with maximum adhesion.

Invoicing

Parallel with the problems of producing 2,000 – 4,000 labels each month (a typical professional magazine circulation), a similar problem arises with invoicing. Subscriptions to journals and professional institutions fall into two main classes – those paid for by individuals and those paid for through agents or companies. In these latter cases, each invoice has to show who receives the journal or membership facilities, and the agent or executive who will authorise payment of the invoice.

As anyone with commercial experience will know, any failure to get invoicing details exactly right provides an excellent excuse for delaying payment.

Therefore any automatic system for producing labels must fit in exactly with the invoicing system: ideally, the same system and equipment must fill both requirements.

Whilst simple addressing equipment will produce labels, the use of a more sophisticated approach allows for the development of a data base which can be used for marketing purposes. It is this skill in data base architecture, design and coding that differentiates the approach described below from a simple addressing machine solution.

For financial and circulation control purposes, it is necessary to know how many subscribers exist (not least to tell the printer the length of the print run), and the publisher to know the trend of additions and deletions. For invoicing purposes, the subscribers need to be batched into periods, with a mechanism for recording paid subscriptions so that an updated reminder list of those unpaid can be produced automatically.

By coding subscribers by country and by section of the industry or profession served, it is possible to answer such questions as "How many potential buyers of Carding Equipment are there in Guatemala?"

The code chosen finally is meaningful when visually inspected, and can be easily used with punch cards or in magnetic format (see below).

03	0272	031	09
Invoicing month (i.e. March)	Individual subscriber Number	Country	Market Segment

Figure 1

Stages in the Solution

(a) Use of a General-Purpose Computer Bureau

In the example used in this article, the publisher elected initially to use the services of a general-purpose computer bureau, having previously decided that his conventional stencil addressing equipment was too costly to use. After a very few periods, the address list was in chaos, the extent of which was made clear when it become necessary to re-create a new current master file from the corrupted computer printout.

Various obvious errors had occurred:-

1. No master file of addresses had been kept by the publisher. This is clearly necessary for commonsense commercial reasons.
2. The computer system required additions, deletions and changes or address to be put in by the publisher's office staff. No-one on his staff was competent to operate a terminal, nor did anyone understand the system commands which have to be used for computer file updating.

The computer bureau had seriously under-estimated the human problems in dealing with this task. The cost of the service was underquoted for the amount of trouble involved, so that the high level of personal service needed to keep the system operational could not be given.

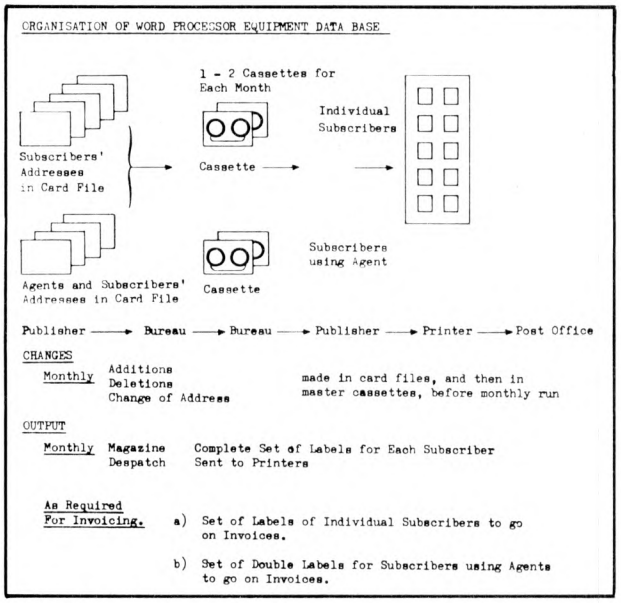


Figure 2

(b) The Second Solution — Use of a Word Processor (see figure 3.)

An initial examination of the problem, when given several corrupted part printouts, showed that it was necessary to create a master data file. The coding system used by the Bureau was not easily interpreted, and the subscribers' addresses were jumbled up in random order. Owing to the use of incorrect inputting routines, many addresses were duplicated.



Figure 3

The Kalle Infotec 7000 Automatic Typewriter installation. Some 2000 labels for the monthly despatch cycle can be produced per 8 hour day, and 950 two label sets for invoicing purposes.

These lists had to be sorted into months in order to relate to annual subscription invoicing procedures, then into countries, and then into individual subscriptions or subscriptions through agents.

To produce the new data base on cassettes took six weeks of very hard work, largely because the written records available were difficult to read. In other cases, where adequate records exist, preparing the cassettes is a simpler operation except where difficult foreign addresses (e.g. Hungarian or Turkish) are involved where a letter-by-letter verification is required.

The operating system is shown in Figure 3, and some technical details are set out in graphic form in Figure 2.

(c) The Third Solution — Use of a Mini-Computer (see figure 4)

A year's experience of operating data base on a word processor showed that the most desirable improvement was in speed of producing the labels. To produce the monthly run of 2,400 labels (in this case) took at least two days in a busy office coping with incoming mail and telephones. It was therefore decided to take advantage of a mini-computer installation, and to write a generalised control program that could be used by any organisation with a computer with a BASIC Compiler.

With a computer it is easier to provide routines for checking the input data. The use of disk storage enables any subscriber's address to be accessed in a fraction of a second. This solution allows the output to be produced in about 2 hours, and this would be far faster if a different type of line printer was available.

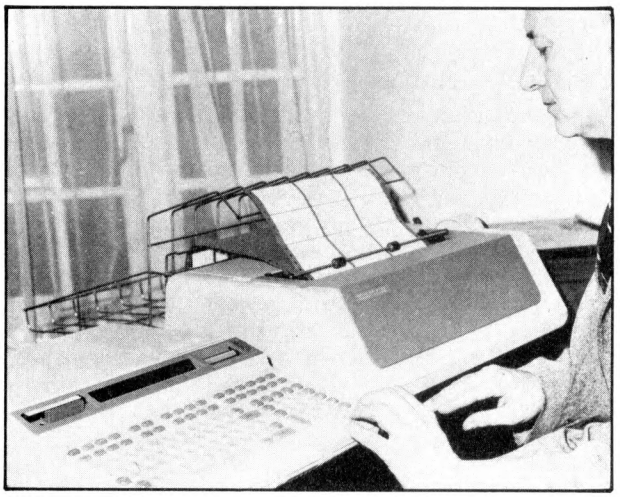


Figure 4

The Hewlett Packard 9825 Mini Computer installation. This solution is capital investment intensive, and requires skilled programming and deft operating. However, the speed of production is about five time faster than with the Automatic Typewriter system, and alterations to subscriber information can be made much more quickly. The system could be speeded up to at least 100% by the use of a different Line Printer.

Summary

The technical solution to this problem has been optimised to a point that any further saving in time would be difficult to justify in relation to the required capital cost. With experience some "simple to use" routines have been established that greatly assist the monthly operating of the program suite.

However, unless there is accurate original data from the publisher, which is kept up-to-date with the use of a simple card file system, and unless the necessary changes are communicated efficiently to the computer operator, no computer, however skillfully handled, will solve the communications problems that arise between producer and customer (in this case, publisher and subscriber).

Since the work described above was done, the availability of such machines as the TRS 80 have greatly decreased the capital cost of such projects. When the Tandy Disk Unit and Line Printer come on the market, the entire hardware cost will be less than £2500. The programs will run on a 16K machine. The main operational problem is fast access to the data files on disk, and until the Tandy Disk Unit becomes available in the UK, we cannot report on the use of the program on this machine, but anticipate no difficulty at all.

HOW TO CONTROL YOUR BASIC FUNCTIONS

Christopher Smith,
Queen Elizabeth College,
Kensington W.8

Infuriating

BASIC can be infuriating: not so much that it is too simple a language but because of the simplicity of the input and output commands, which results in inflexibility. I find BASIC an extremely pleasant language but surely nothing turns off computer *users,* as opposed to programmers, more than printouts like:

COST = £ 25. 354724

when anyone knows that even a penny is virtually valueless. Or how about the interactive session which proceeds:

OPTION 1 or 2 ?

and following the sensible reply of the tired user, NO, the computer insists on:

REDO FROM START

or some such inexplicable advice; and continues to refuse any sensible message the poor fellow types in. This, you or I know, is because the INPUT statement must specify numeric or character string (abbreviated to an unlovely 'string' and donated by $ sign) input. Must ? Not if you were sensible enough to operate a system using extended BASIC, for VAL and STR $ commands interconvert strings and numbers in a millisecond. So why not:

```
PRINT "OPTION 1 OR 2";
INPUT X$: X = VAL(X$)
```

and any string entry will give X = 0 to which you can take appropriate action.

Approachability

Commodore's PET 2001 must be classed as the most approachable computer on the market. All that a fool (sorry, 'a user') need know is how to insert a cassette and operate the key 'RUN' (unfortunately an upper case key). Then, if the programme is good, the screen tells the user what to do. But consider her plight when in response to:

ENTER YOUR AGE ?

she simply pushes the RETURN key. She gets a flashing light (a null input to PET's Microsoft BASIC drops the programme and returns to command mode). Then following her reluctant entry of 20 — you should be so lucky — not even the supervisor can make the beastly programme work. She has obliterated line 20.

Peeking and Poking

Solving this problem is less easy. Either you must alter your BASIC or PEEK and POKE into its book-keeping stores. For the latter case it is best to make a subroutine for all your keyboard entry operations. The following subroutine, written for the PET 2001, allows normal numeric entry, shields against null entries and interprets alpha entry as a single letter to immediately redirect (i.e. without need of the return key) to a new part of the programme. This is the normal need for most number crunching programmes where number entry is ideally stopped at the user's wish to allow editing (E), data display (D), to get an answer (=) or even abort (A) the programme. It is written for the PET 2001: the first line simply provides a flashing cursor and checks for any keyboard entry.

```
1000 PRINT"?"; :X=TAN(TAN(.3)):PRINT"←↩"; :X=
TAN(TAN(.3)) :IF PEEK(SN) =0 THEN GOTO 1000
1005 SC=PEEK(SB):IF(SC>SL)AND(SC<SU)THEN INPUT
X: RETURN
1010 POKE(SN,0): IF SC =SZ THEN PRINT "NULL
INPUT"; : GOTO 1000
1015 ON SC + 1 – SA GOTO 1100, 1200,1300,1400,1500,
1600 etc.
1020 PRINT" INVALID ENTRY": GOTO 1000
1100 REM SUB FOR = A
1199 RETURN
1200 REM SUB FOR B
etc.
```

← means cursor backspace.

Data Entry

'SN and SB are the locations of the number of characters in the input buffer and of the first character of the input buffer (for the PET SN = 525 and SB = 527). SZ is the buffer code for 'return' (PET used ASCII here: SZ = 13), SL, SU and SA are the codes for the lower and upper range of valid numeric items and for the bottom end of the list of alpha characters (e.g. SL = 106 SU = 122 SA = 64). It may seem rather odd at first to have to use a subroutine for data entry but the sooner you start the less the problems.

Limit Routines

Rather more commonly used are routines to limit the output statement of BASIC. The simplest is to restrict to an integer format:

```
PRINT INT(X + 0.5)
```

The INTEGER statement always rounds down – so it is up to you to compensate for this. Likewise to print to a fixed number of decimal places, e.g. to avoid quoting millionths of a penny:

```
PRINT INT( X+10↑N + 0.5)/10↑N
```

where N is the required number of decimal places (avoid the exponent expression if you are sure in advance of the value of N to use).

Quite often one wants to limit the total number of digits rather than just the number after the decimal point. Here the problems start: how does one restrict the number 6789 to three figures? The usual solution is to print *** – presumably on the principle 'garbage in, garbage out'. The following routine prints a large number in its (integer) entirety. It limits printout to 4 significant figures: note the need for a zero check.

```
100 M=1 : IF X=0 GOTO 110
105 IF ABS(X)<1000 THEN X=X*10 : M=M*10 : GOTO
                                          105
110 PRINT INT(X)/M
```

Established computer users (there is an inclusive OR between 'computer' and 'users') seem to feel there is a need to specify the output format in a manner akin to that of FORTRAN where the field size as well as the number size is controlled. In the following programme, which needs the LEFT$, RIGHT$ and STR$ statements, the number of digits before the decimal point is P, those after is Q and the number of spaces following is R. Thus the total field size is P+Q+R+2 (1 each for the sign and point). First it is necessary to set up a blank character string of suitable length for your requirement:

```
5 B$= "                    "
```

then the subroutine:

```
200 IF X>= 10↑P THEN PRINT "****" + RIGHT$(B$,
P+Q+R–2);: RETURN
205 PRINT LEFT$(STR$(X),1);: X=ABS(X):
PRINT RIGHT$(B$+STR$(INT(X)),P);: X=X–INT(X)
210 IF X=0 THEN PRINT RIGHT$(B$,Q+R+1);: RETURN
215 PRINT "." + RIGHT$ (STR$ (INT(10↑Q*(X+1))),Q)
+ RIGHT$(B$,R);: RETURN
```

Whilst this series of routines is neither complete or even totally correct (some of you may have noticed some overflow checks that would be required in a fully professional version of these routines) they should at least show that there is no need to restrict oneself to the very BASIC input and output modes.

224 ARTICLES

Kilobaud has more articles than any other microcomputer magazine. During 1977, for instance, there were 224 articles in Kilobaud—880 pages of articles—that's like a very large encyclopedia of microcomputing. There will be even ... more articles in 1978.

FOR THE BEGINNER

Though Kilobaud covers both the technical and programming sides of microcomputers, each article is written with the beginner in mind. No other magazine makes it as easy to understand microcomputers.

THE BEST PROGRAMS

Only Kilobaud offers programmers both the publication of their programs in the magazine plus a large royalty for the program if it's issued on cassettes. Is it any wonder all of the really good programs are being published in Kilobaud? You'll find the best in games, diagnostics, teaching, music, business, etc., programs in Kilobaud. We're very heavy on programs.

HOBBY OR BUSINESS?

Both businessmen and hobbyist want the same thing: to understand microcomputers ... and this is the purpose of Kilobaud. KB also has articles aimed at the businessman to help him know what is available and what it will do.

THE PUBLISHER

Kilobaud is published by the same people who put out 73 Magazine —the largest of the amateur radio magazines.

MONTHLY COLUMNS

Keep up with the latest developments and readers' programs through the letters column ... a KIM column ... one on the TRS-80 ... a column on BASIC ... etc. The editorials will keep you up to date on money-making ideas, the progress of the industry and more. The New Products column is particularly useful, including the results of tests of the latest equipment in the Kilobaud microcomputer laboratory—the most complete in the industry.

SUBSCRIBE TO KILOBAUD

You can get Kilobaud fast. Copies are flown to Europe immediately upon publication and mailed directly from the United Kingdom, so your copies are current. Send subscription orders for United Kingdom and Europe, £20 sterling per year, to:

If you only read ONE American computer magazine...

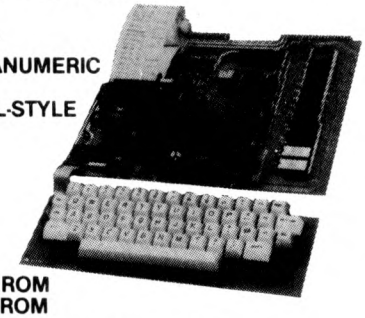
it should be

The Small Computer Magazine kilobaud

L. P. Enterprises
313 Kingston Road, Ilford, Essex. 01-553 1001
Barclay Card, VISA, Diner's Club and American Express, Access Card honored.

AIM 65

£249·50 + VAT



FEATURES INCLUDE:
* 20 COLUMN PRINTOUT
* 20 CHARACTER ALPHANUMERIC DISPLAY
* FULL 54 KEY TERMINAL-STYLE KEYBOARD
* TTY INTERFACE
* TWIN CASSETTE INTERFACE
* RAM — 1K TO 4K OPTIONS

OPTIONS INCLUDE:
8K BASIC INTERPRETER ROM
4K ASSEMBLER/EDITOR ROM

AIM 65 comes to you fully built and tested with a full alphanumeric keyboard, 20 character display and a 20 column printer — for keeping a permanent record of all your work. Available in 1K- and 4K-byte RAM versions, AIM 65 is designed around the 6502 CPU, which has 64K address capability with 13 addressing modes. This is the microprocessor at the heart of many other, more costly, systems such as PET and APPLE.
AIM 65 has a 4K ROM-resident monitor program for all peripheral control and user programming functions.
Spare sockets are included for expanding on-board program memory via user PROM-based programs and/or Rockwell assembler, text editor and BASIC interpreter plug-in options.
AIM 65 has a connector for external access to system bus for memory and I/O expansion, a separate connector for interfacing a teletype and two cassette recorders. There is a user-dedicated Versatile Interface Adaptor, featuring three 8-bit, bidirectional ports (two parallel, one serial) and two 16-bit interval timer/event counters — thus allowing the user to interface his own system, without extra interface devices in many cases.
AIM 65 is probably the most effective, low-cost microcomputer development system available — an invaluable educational aid to first time users and and ideal general purpose microcomputer for the engineer.
AIM 65 is available in the UK only from PELCO ELECTRONICS LTD at £249.50 + VAT, complete with User's Manual and Schematic, R6500 Programming and Hardware Manuals and a handy pocket reference card.

Pelco (Electronics) Ltd
Enterprise House,
83-85 Western Road, Hove, Sussex BN3 1JB
Telephone: Brighton (0273) 722155

Buy it with your Access or Barclaycard.

How To Enter Computer Programming

George Penney NCC Careers Projects Manager & Director of the NCC Threshold Scheme
NCC, Oxford Road, Manchester M1 7ED

A substantial number of employers labour under the delusion that none but graduates can be expected to become successful programmers. The subject or subjects of the degree are of little concern; the possession of any degree is regarded as a certificate of intellectual ability. There is, in some organisations, also the problem of status; they have never taken non-graduates for programming and therefore all their programmers have the status accorded to graduates, in whatever capacity such graduates might work. It follows that, in order to be acceptable to the whole range of possible employers, the wisest thing is to get a degree. Other employers – one might almost say all other employers – will not consider anyone for programming unless they have either A-levels or programming experience. Since this implies that there is no way of getting programming experience without already having at least A-levels, this seems to tie up the whole question; stay at school and take A-levels, then either apply for jobs or apply to University, or both.

Supposing, however, that your A-levels are not of a great enough number or a high enough grade to allow you to enter on a degree course, what then? Should you accept the conventional wisdom of the employers and conclude that you are not clever enough to become a programmer? Now, the possibility exists that you are not; but then the possibility or rather probability exists that someone with A-levels or a degree is not suited to programming either; and it is not just a question of being clever; being mentally suited to the job is what really counts. If you are not so suited you will not be a good programmer, and it almost certainly follows that you will not enjoy programming, either. From the studies we at NCC have carried out, I would estimate that about one in ten of the population has the ability and the other mental characteristics that go to make a satisfactory programmer; which is far from meaning an outstanding programmer, which, at a guess, applies to no more than one in a thousand of the population.

Let us suppose, then, that you achieve only one A-level and that not a very good grade. Most Polytechnics will be prepared to consider you for entrance to their Higher National Diploma (HND) in Computer Studies.

Strange as it may seem, many employers prefer holders of this qualification to degree-holders. This is because there is much more opportunity on the HND course to do practical work, which most employers need done, than on most specialised computing degree courses – and, very obviously, much more than any other degree course.

What if you get no A-levels at all? You can try writing scores of letters, and you might land a job as a computer operator. Once you are in a computer installation you have a sporting chance of convincing your manager that you could make a programmer and of being offered appropriate training at the firm's expense. There is the slight problem, though, that computer operators, with shift allowance (many operators are required to work shifts) and possibly overtime as well, generally take home more money than a trainee programmer; so if you choose this route you have got to be prepared to take a drop in salary when you make the change.

Support Your Local MICROPROCESSOR!

With our MM57109 Based Kit, add all the functions of a powerful scientific calculator to your Z80 or SC/MP based computer.

* Supplied complete with instructions and software to interface to NASCOM I Kit. No modifications to the NASCOM I are required.
* May require as little as 100(H) Bytes of ROM/RAM to operate.
* Your Microprocessor is left free to organise input and output of Data.
* Gives you direct control of computer programs – No need to work via high level languages.

COST: £29.75 + £1.25 p.p.

ONLY FROM:

CONTROL AND ELECTRONIC DEVELOPMENTS

BOX

PET 2001 Personal Computer with 8K Ram, little used and for sale because Company must expand to a much bigger system. Offers over £400.
Tel: (0291) 690484.

TRS-80?
IF YOU HAVE the machine, we have the software BUSINESS/FINANCE/MATHS/GAMES etc. Send SAE for list to GENERAL SOFTWARE 15 Rockwood Avenue Crewe

"MAKE A START: 8080A CPU £6
IMPROVE YOUR MEMORY: 2102 £1.75
(8 for £13.50)
VERO DIP BREADBOARD 2.78 Wiring Pen £2.63
BOOKS: Introduction to Basic £6.50
Starship Simulation £5.10

Data with all purchases.
ACCESS welcome by phone or mail.

TORTSTAR LTD ELECTRONICS
14 Robert Rayner Close
Orton Longueville
Peterborough Cambs PE2 0AY

TELETYPE 33ASR's

Used TTT's available capable of 20MA or RS232 working.
Each machine has 30 Day Return to Depot Warrantee.
Large Stocks Available to Trade Enquirers.
£250 each, collected London.
0634-362823 Chris Turner.

Another way – but only if the aspirant concerned happens to live on Merseyside – is to apply to join the special non – A-level batch of entrants to the HND course. This is about to happen for the first time, as a result of the staff at Liverpool Polytechnic reading of the NCC Threshold Scheme, of which more in a moment. I was approached by the Polytechnic people to help them in sifting applications from non – A-level people to join the experimental intake to which they had secured assent in principle. By using a battery of tests, we were able to identify some 15 out of 29 applicants who not only had the intellectual ability to follow the course but also showed signs of having the other characteristics which make a satisfactory programmer. If this experiment is successful – and I am very confident that it will be – there can be no doubt that Liverpool will repeat it and that other Polytechnics will also start dipping their toes in the water. So, by the time that Master Kallenbach – (whose enquiry sparked off this article) has taken A-levels, the London polytechnics also may be offering this method of entrance.

But must you stay on at School and take A-levels to have any real chance of becoming a programmer? At present, this is not necessarily so. The answer is the NCC Threshold Scheme. This is open to any school-leaver, between the ages of 16 and 19, who is either unemployed or in a dead-end job, provided that the tests and interview which we set up, in the various colleges running the scheme, show a good probability of success on the course and in subsequent employment. The course, administered by NCC and funded by the Manpower Services Commission (MSC) lasts for 40 weeks, of which 16 are in college and 24, split into two or three periods, doing practical work in a computer department. So, anyone who has simply had enough of school at 16 but still believes himself, or herself, cut out for computer programming, has only to write to me for a leaflet and an application form. In London, the chances of getting on the course are very high, because there are only two or three applicants for each course place, as opposed to most other parts of the country, where there are between 5 and 10 applicants for each place.

I must not leave the subject of entry to programming without mentioning the MSC Training Opportunities Scheme (TOPS). Supposing you have left school and not managed to get on a Threshold Course? So you search and find some other job, I hope, rather than simply joining the dole queue. Once you have been out of full time education for 3 years you are eligible to apply for a TOPS course – which could be in programming or almost any other vocational subject. You will still be required to take tests, but your abilities could have changed in those 3 years or the competition might be less fierce at that time or in that part of the country.

Those, to the best of my knowledge, are the only ways of becoming a computer programmer without spending a substantial amount of your own money. There are commercial computer schools, some good, and some not so good. The good ones will provide an excellent training and help in finding a job afterwards; the bad ones will take your money and give you something that they call a training course, accompanied by a worthless certificate and many meaningless promises. If you do come to this as a last resort, write to me for a list of the commercial computer schools which subscribe to the NCC Code of Practice. This will not guarantee a good course or a job at the end, but it will, at least, guarantee that the intentions of the school are good and that they will stand by any promises made.

One final word: if you decide that you want to be a computer programmer – or anything else for that matter – keep trying, never give up, and the chances are that you will succeed.

Learning Geography With a Computer



John D. Lee and Timothy D. Lee

Some aspects of education still involve considerable amounts of learning by rote, particularly where large amounts of factual knowledge have to be assimilated rather than an understanding of concepts. Computer assisted learning may be effectively employed for this purpose. If an element of challenge is introduced – can you get a better score than your friend, or a better score today than yesterday, – then tedium and drudgery are replaced by an element of fun.

The program described is written in BASIC for interactive use on a computer terminal. It tests the user's knowledge of countries and their capital cities, and one may choose to be tested either on one particular continent, or on the whole world.

For variety the computer asks questions in one of two ways – either "WHAT IS THE CAPITAL OF ?", or "WHAT COUNTRY IS THE CAPITAL OF?" The choice of which way the question is asked is determined randomly by the computer, but one (country, capital) pair of data is only used once in a test. The order of the questions with a test is also determined randomly, so that it is impossible to learn the correct sequence of answers. If a question is asked using for example the third data pair (PORTUGAL & LISBON), then the third element of the L array is changed from 0 to 1. Repetition of questions on the same data pair is avoided by only asking questions on data pairs whose corresponding element in L is zero.

When the user has typed in an answer to a question, the computer replies "CORRECT" or "WRONG". If the user does not know the answer, he may type PASS. If you have passed, or the answer was wrong, then the correct answer is printed out. The computer then continues with the next question, and so on, until all the data pairs for the continent chosen have been used. At this point the score is printed out as the number of questions right, wrong and passed. The user is then asked if he would like to continue. At any stage you may finish the test by typing QUIT, when your current score is given.

The data pairs are stored at the end of the program in DATA statements which are arranged in continents. When a continent is chosen, the variables L and U are set to define the lower and upper limits of the data pairs for this continent. For example, for America L=51 corresponding to CANADA, OTTAWA whilst U=62 corresponds to VENEZUELA, CARACAS. If users wish to add more data pairs for themselves, they must be placed in the appropriate block of continent data, and the values of L and U for this and subsequent continents adjusted appropriately.

With suitable modification to the questions and data, the program could be adapted to handle mountains and countries, Kings and dates, battles and dates, chemical names and their common names, authors and novels and numerous other similar applications. A challenge to computer addicts is to produce a single program incorporating tests on all of these, which stores the questions and data for each subject in a separate data file which may be accessed by the program.

```
PROGRAM TO TEST GEOGRAPHICAL KNOWLEDGE
======= == ==== ============ =========

TYPE EUROPE, AFRICA, ASIA, AUSTRALIA OR AMERICA TO INDICATE
THE REGION TO BE TESTED. ALTERNATIVELY TYPE WORLD FOR A
GLOBAL TEST
? AMERICA
IF YOU DO NOT KNOW THE ANSWER TO A QUESTION TYPE PASS
TO FINISH THE TEST TYPE QUIT
YOU WILL BE GIVEN THE SCORE EITHER WHEN YOU HAVE FINISHED
A CONTINENT, OR WHEN YOU QUIT.

WHAT COUNTRY IS RIO DE JANEIRO THE CAPITAL OF
? BRAZIL
CORRECT

WHAT IS THE CAPITAL OF VENEZULA
? PASS
THE CORRECT ANSWER IS CARACUS

WHAT IS THE CAPITAL OF PARAGUAY
? MONTEVIDEO
WRONG - THE CORRECT ANSWER IS ASUNCION

WHAT COUNTRY IS SANTIAGO THE CAPITAL OF
? CHILE
CORRECT

WHAT COUNTRY IS LIMA THE CAPITAL OF
? PERU
CORRECT

WHAT COUNTRY IS MONTEVIDEO THE CAPITAL OF
? URUGUAY
CORRECT

WHAT COUNTRY IS BUENOS AIRES THE CAPITAL OF
? QUIT

YOUR SCORE IS 4 RIGHT, 1 WRONG, AND 1 PASS(ES) OUT OF 6 TURNS

WOULD YOU LIKE TO CONTINUE (YES/NO)
? NO
END OF PROGRAM
```

```
10 DIM A$(20), B$(20), C$(20), L(100)
20 PRINT "PROGRAM TO TEST GEOGRAPHICAL KNOWLEDGE"
30 PRINT "======= == ==== ============ ========="
40 PRINT
50 MAT  L = ZER
60 PRINT "TYPE EUROPE, AFRICA, ASIA, AUSTRALIA OR AMERICA TO INDICATE"
70 PRINT "THE REGION TO BE TESTED. ALTERNATIVELY TYPE WORLD FOR A"
80 PRINT "GLOBAL TEST"
90 INPUT A$
100 LET L = 1
110 LET P = R = W = 0
120 IF A$ = "WORLD" THEN 280
130 LET U = 24
140 IF A$ = "EUROPE" THEN 290
150 LET L = 51
160 LET U = 62
170 IF A$ = "AMERICA" THEN 290
180 LET L = 24
190 LET U = 41
200 IF A$ = "ASIA" THEN 290
210 LET L = 42
220 LET U = 50
230 IF A$ = "AUSTRALIA" THEN 290
240 LET L = 63
250 IF A$ = "AFRICA" THEN 280
260 PRINT "REPLY '"; A$; "' NOT UNDERSTOOD. RETYPE CORRECTLY"
270 GOTO  90
280 LET U = 72
290 PRINT "IF YOU DO NOT KNOW THE ANSWER TO A QUESTION TYPE PASS"
300 PRINT "TO FINISH THE TEST TYPE QUIT"
310 PRINT "YOU WILL BE GIVEN THE SCORE EITHER WHEN YOU HAVE FINISHED"
320 PRINT "A CONTINENT, OR WHEN YOU QUIT."
330 FOR I = 1 TO (U + 1 - L)
340 PRINT
350 LET X = L + INT((U - L) * RND(0) + 1)
360 IF L(X) = 1 THEN 350
370 LET L(X) = 1
380 FOR J = 1 TO X
390 READ A$, B$
400 NEXT J
410 IF RND(0) > .5 THEN 450
420 PRINT "WHAT IS THE CAPITAL OF "; A$
430 GOSUB  660
440 GOTO  500
450 LET C$ = A$
460 LET A$ = B$
470 LET B$ = C$
480 PRINT "WHAT COUNTRY IS "; A$; " THE CAPITAL OF"
490 GOSUB  660
500 RESTORE
510 IF C$ = "QUIT" THEN 530
520 NEXT I
530 PRINT
540 PRINT "YOUR SCORE IS"; R; "RIGHT,"; W; "WRONG, AND"; P; "PASS(ES)";
550 PRINT " OUT OF"; R + W + P; "TURNS"
560 PRINT
570 PRINT "WOULD YOU LIKE TO CONTINUE (YES/NO)"
580 INPUT C$
590 IF C$ = "YES" THEN 40
600 IF C$ = "NO" THEN 630
610 PRINT "REPLY '"; C$; "' NOT UNDERSTOOD. RETYPE YES OR NO"
620 GOTO  580
630 PRINT "END OF PROGRAM"
640 STOP
650 REM *** SUBROUTINE TO CHECK THE ANSWERS GIVEN TO THE QUESTIONS
660 INPUT C$
670 IF C$ = "QUIT" THEN 780
680 IF C$ <> "PASS" THEN 720
690 LET P = P + 1
700 PRINT "THE CORRECT ANSWER IS "; B$
710 GOTO  780
720 IF C$ = B$ THEN 760
730 PRINT "WRONG - THE CORRECT ANSWER IS "; B$
740 LET W = W + 1
750 GOTO  780
760 PRINT "CORRECT"
770 LET R = R + 1
780 RETURN
790 DATA "ENGLAND", "LONDON", "FRANCE", "PARIS", "PORTUGAL", "LISBON"
800 DATA "SPAIN", "MADRID", "ITALY", "ROME", "SWITZERLAND", "BERN"
810 DATA "EAST GERMANY", "BERLIN", "WEST GERMANY", "BONN", "BELGIUM"
820 DATA "BRUSSELS", "DENMARK", "COPENHAGEN", "NORWAY", "OSLO", "SWEDEN"
830 DATA "STOCKHOLM", "FINLAND", "HELSINKI", "IRELAND", "DUBLIN"
840 DATA "POLAND", "WARSAW", "CZECHOSLOVAKIA", "PRAGUE", "AUSTRIA"
850 DATA "VIENNA", "HUNGARY", "BUDAPEST", "YUGOSLAVIA", "BELGRADE"
860 DATA "RUMANIA", "BUCHAREST", "BULGARIA", "SOFIA", "GREECE", "ATHENS"
870 DATA "HOLLAND", "THE HAGUE", "TURKEY", "ISTANBUL"
880 DATA "SYRIA", "DAMASCUS", "LEBANON", "BEIRUT", "ISRAEL", "TEL-AVIV"
890 DATA "JORDAN", "AMMAN", "IRAQ", "BAGHDAD", "PERSIA", "TEHRAN"
900 DATA "SAUDI ARABIA", "RIJADH", "PAKISTAN", "ISLAMABAD", "INDIA"
910 DATA "DELHI", "BANGLADESH", "DECCA","CHINA","PEKING","USSR","MOSCOW"
920 DATA "VIETNAM", "HANOI", "CAMBODIA", "PNOM PENH", "THAILAND"
930 DATA "BANGKOK", "JAPAN", "TOKYO", "THE PHILIPPINES", "MANILA"
940 DATA "WESTERN AUSTRALIA", "PERTH", "NORTHERN TERRITORY", "DARWIN"
950 DATA "QUEENSLAND", "BRISBANE", "SOUTH AUSTRALIA", "ADELAIDE"
960 DATA "NEW SOUTH WALES", "SYDNEY", "AUSTRALIA", "CANBERRA","VICTORIA"
970 DATA "MELBOURNE", "TASMANIA", "HOBART", "NEW ZEALAND","CHRISTCHURCH"
980 DATA "CANADA", "OTTOWA", "U.S.A", "WASHINGTON", "BOLIVIA", "LA PAZ"
990 DATA "BRAZIL", "RIO DE JANEIRO","URUGUAY","MONTEVIDEO","PERU","LIMA"
1000 DATA "ARGENTINA", "BUENOS AIRES", "CHILE", "SANTIAGO", "PARAGUAY"
1010 DATA "ASUNCION", "COLOMBIA", "BOGOTA", "ECUADOR","QUITO","VENEZULA"
1020 DATA "CARACAS"
1030 DATA "EGYPT", "CAIRO", "LIBYA", "TRIPOLI", "ETHIOPIA","ADDIS ABABA"
1040 DATA "SOMALIA", "MOGADISHU", "KENYA", "NAIROBI", "UGANDA","ENTEBBE"
1050 DATA "ANGOLA", "LUANDA", "SOUTH AFRICA", "CAPE TOWN", "RHODESIA"
1060 DATA "LUSAKA", "TUNISIA", "TUNIS"
1070 END
```

PCW OPEN PAGE

The Amateur Computer Club View

Mike Lord



An encouraging development for the computer hobbyist with limited funds is that groups are now starting to form around educational establishments which have hardware available for use as well as enthusiastic lecturers willing to explain the mysteries of DIY computing. For example, the *North London Hobby Computer Club*, which will have had its inaugural meeting by the time this appears in print, has been founded at the Polytechnic of North London by senior lecturer Robin Bradbeer, while Diane Hayes at *Corby Technical College* has started a computer club with access to the college's three Research Machines' 380Z microcomputers, and plans for workshops and courses in Z80 machine code and BASIC programming. *Liverpool University* have also offered their facilities to ACC members who may be interested in getting together on a regular basis to discuss and develop microcomputer hardware and software – contact Malcolm Taylor on 051-709-6022 ext 2955 for further details.

Also in *Liverpool*, the Merseyside Mini/Micro Computer Club has organised a talk and demonstration on the NASCOM 1 for 16th November, and are arranging a further meeting on 8th December. Anyone wishing to attend should get in touch with Martin Beer, c/o STEM, 19 Abercromby Square, PO Box 147, Liverpool University.

Computer enthusiasts living in the *West Midlands* are invited to contact John Tracey, 100 Booth Close, Crestwood Park, Kingswinford DY6 8SP (telephone 0384 70097), as he would like to see a local group in that area.

For those near *Bristol* and/or *Cardiff*, Pete Hesketh on Shire Newton 596 (evenings and weekends) wants to help spread personal computing along the banks of the Severn.

M. Alexander, of 5 Brattelby Crescent, *Lincoln* LN2 2EB hopes that there are enough enthusiasts living near him to support a local micro club. Anyone interested should write to the above address or telephone Lincoln 23084.

Following my effusive articles in recent issues of PCW, Pete Harris, of 119 Carpenter Way, Potters Bar, Herts EN6 5QB has offered to start a 'personal' *PDP/LSI 11 User Group* to promote the interchange of ideas and expertise. He would like to hear from anyone using an '11 of any size running under any operating system.

Finally, for those who may be wondering what has happened about the proposed re-examination of cassette tape standards for amateur use (see September's PCW, page 48), Alan Secker reports that he has been overwhelmed by the response but is currently planning to stage the great debate early in the new year.

A letter from Tandy

Further to our telephone conversation I wish to confirm that we are looking for Software to operate on the TRS-80 Microcomputer System and we would be willing to exchange on a two-for-one basis for an original computer programme written for the TRS-80.

We hope to collate and produce a TRS-80 Software Library and we would of course welcome any programmes for entry.

John Bullock, Computer Manager,
Tandy Corporation (Branch UK), Bilston Road,Holyhead Road,
Wednesbury, Staffordshire WS10 7JN
Telephone 021-556 6101/2, Telex: 339423

MERSEYSIDE MINI/MICRO COMPUTER CLUB
(from P. Broderick)

We have just started a club for users of mini and microcomputer users on Merseyside under the guidance of Science and Technology Education on Merseyside (STEM).

An inaugural meeting was held one evening during the Merseyside Science and Technology Fair and considerable interest was expressed by a wide cross-section of the Merseyside educational and industrial community, including many who are not at present computer users.

A number of local distributors have promised help, both in giving talks, and demonstrating their products as well as helping in publicising the Club's activities.

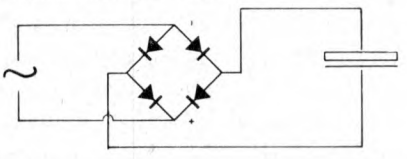
MEETINGS TO DECEMBER, 1978

Thursday 16th November.	A talk and demonstration on NASCOM 1.
Friday 8th December.	To be arranged.

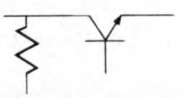
Address: 19 Abercromby Sq., P.O. Box 147, Liverpool University, Liverpool L69 3BX.

Corrections to PUNCHLINES (July, No.3)

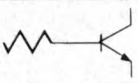
PSV Rectifier should be as follows:–



ASV Transistor, missing the emitter arrow:



Motor relay cct transistor, also missing the arrow



Logic diagram; 'CIR' in all cases is wrong, should be CLR, abreviation for 'CLEAR'

Block Diagram; Data outputs 0 → 7, not 0 → 77

LOW COST EXPANDABLE TTY, VDU, I/O BASIC, LIMBUG

LMC6800-2 microcomputer

* EXORciser is the trade mark of Motorola Inc.



EXORciser·BUS*

LMC 6800-2 IS VERSATILE

This microcomputer uses the same pin out as the EXORciser Bus by Motorola. And, all cards, from various vendors, for the EXORciser will fit the LMC 6800-2 System with little or no modifications. So you are not dependent on a single-source supplier for future expansion.

LMC 6800-2 HAS BUILT-IN LOGIC ANALYSER

A unique "TRAP" facility provides you with a built-in Logic Analyser at no extra cost. The "trap" lets you examine every single machine cycle- not just instruction cycles - in running your programme. In conjunction with the powerful and comprehensive LIMBUG monitor, you can use the "trap" for debugging of hardware and software and even go "backwards" in time to see exactly how the programme was executed before it crashed. LIMBUG will let you add comments, single step the CPU, insert "breakpoints" during programme run and move blocks of programme !.!.

LMC 6800-2 SPEAKS BASIC

The 8K ROM-based BASIC Interpreter for the LMC 6800-2 microcomputer is almost foolproof!. You cannot "crash" it, unlike most other RAM-based BASIC Interpreters you have probably seen before. It' s a "full" BASIC, with floating point arithmetic and numerous other facilities. Ideal language for use in schools for computer science studies, or as a powerful schools computer.

LMC 6800·2 £370*

4K RAM, 1K ROM LIMBUG,
8K BASIC INTERPRETER IN ROM,
Teletype/VDU Interface and "Trap",
19" Rack and 21-slot Motherboard,
All necessary Power Supplies and documentation.

* Kit form. VAT and carriage extra.

limrose

LIMROSE MICROPROCESSOR DIVISION,
241-243 MANCHESTER ROAD,
NORTHWICH, CHESHIRE, CW9 7NE.
TEL. 0606 41696/7

PERSONAL COMPUTER WORLD

OUR SHOW



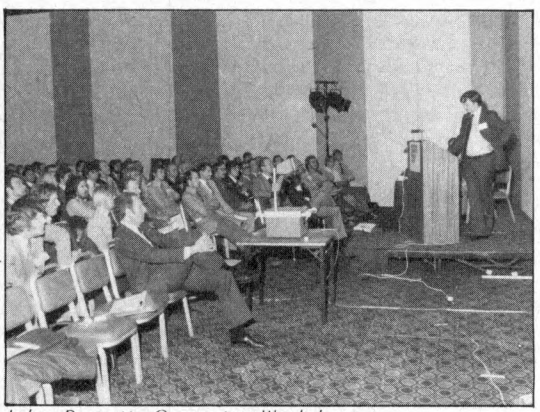
Brisk business: 1.



Brisk business: 2.



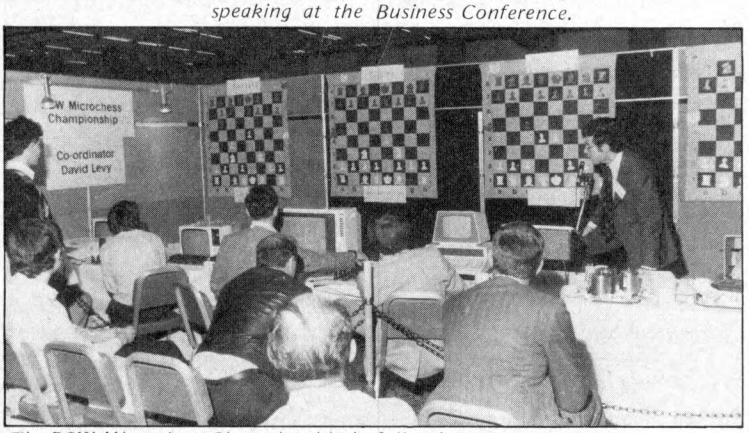
Brisk business: 3



John Burnett, Computer Workshop, speaking at the Business Conference.



Features area: Brian Reffin Smith and his MARTA program filmed by London Weekend Television.



*The **PCW** Microchess Championship in full swing.*

PCW Book Review



Michael James

AN INTRODUCTION TO MICROCOMPUTERS

Volume 1 Basic Concepts Adam Osborne, 1976; 280 pp. (Adam Osborne and Associates Inc., 5¼" x 8", £5.95)

Most personal computer people regard the microprocessor as simply a small minicomputer, with not quite so much (enough?) power. Why it doesn't have so much power is taken to be a matter of economics and technology, certainly not design. In short, an MPU (MicroProcessor Unit) is taken to be a single chip CPU. However, the view taken by the electronics and manufacturing industries is very different. The MPU is seen as a single element which can be used to replace *any* control system. The words control system may seem to make the MPU of fairly limited use, but they cover a very large set of applications. It is, *in theory,* possible to make a radio, an amplifier, an egg timer, an oscillator, a transmitter, etc., etc., from an MPU. Thus the manufacturer's dream may yet come true — a production line manufacturing one product, an MPU, its function being set much later by "programming" it. From this point of view the computer, mini or micro, is simply one of the products covered by the powerful microprocessor concept along with musical doorbells, etc.

An Introduction to Microcomputers, Volume 1, takes this special products approach. It explains, in detail, why current MPU's have the features they have. However, because of the approach, some of the explanations may seem irrelevant to the computer hobbyist.

For example. "Whatever you do today becomes a ROM chip and will never again change." or "While this justification for indirect addressing makes a lot of sense in complex minicomputer applications, it makes absolutely no sense in microcomputer applications." Most personal computer programs do not end up in ROM and these days anything that makes sense for a minicomputer is good for a micro. To continue in this vein would be to completely misrepresent the aim of this book. In seven chapters it condenses all the knowledge needed to understand a modern MPU and, more importantly, why they are as they are.

Chapter 1 introduces the microprocessor. Chapter 2 deals with the usual necessary theory — number systems, arithmetic and Boolean algebra — and is no better and no worse than any of the other introductions. Chapters 3 and 4 deals with the hardware design of an MPU chip. This is covered in more detail than is usual. A feature that again makes it clear that this book is not aimed at the computer hobbyist is the section on "Bit Slice Processors".

Chapter 5 is titled "Logic Beyond the CPU" and deals with peripherals and general communications with the outside world. The section on serial 1/0 is excellent. The last two chapters seem to be about software, being entitled "Programming Microcomputers" and "An Instruction Set". However, the hardware still dominates. The choice of each addressing mode, input/output instruction, etc. is considered in terms of how it will affect the hardware.

"An Introduction to Microcomputers" is a little expensive for its size but not for the abvious amount of care and work that has gone into writing it. If you need a grounding in MPU's from the *hardware* point of view without dealing with a *particular* chip, then this is an excellent book.

MICROPROCESSORS: NEW DIRECTION FOR DESIGNERS

Ed. E. A. Torrero, selected from Electronic Design, 1975; 135 pp. (Hayden Book Co. Inc., 8¼" x 11", £7.00)

This is a collection of reprints from Electronic Design spanning 1973 to 1975. Some, but by no means all, of the articles are outdated. The survey articles list and deal with only some of the microprocessors around today, but the comments are still true and worth reading. The applications articles are far from outdated and are interesting and well written.

The book is aimed mainly at the professional reader but there is quite a lot for the enthusiastic amateur. There is little about software and nothing about complete systems but a lot of discussion and some hardware ideas.

"Microprocessors" is divided into four sections. The first is a general overview of the situation before 1975. Section two is titled "Getting Started: Microprocessor Basics" and includes items such as "Traffic light controller: an example" and "Analysis of microprocessor instruction sets". It consists of a collection of discussions of the pros and cons and problems of using microprocessors. The last two sections are the least dated and most useful, dealing with extending microprocessors by external hardware and applications. There is a description of an add-on hardware multiplier and an interrupt handler in section three. The applications covered in section four are printer control (6800), simplifying peripheral controllers (6800), PLL for motor control and microprocessors in (measuring) instruments (4004). They are all interesting but too few in number.

There have been many articles in various magazines/journals/books dealing with more adventurous applications and more extensive surveys of what is available since this collection was published. It is a good introduction and covers some odd topics difficult to find elsewhere but nowadays hardly lives up to its subtitle.

FOR SALE

NEVER USED, KIM 1 WITH MOTHERBOARD AND TWO 8K MEMORY BOARDS. ONE MICROSOFT 6502 BASIC SET. SURPLUS PARTS FOR 18 AMP POWER SUPPLY ONLY £475. RING J. COLEMAN, MAIDENHEAD 34362. 9 ALTWOOD ROAD.

BSEA

PARTNERSHIP IN MAYFAIR EMPLOYMENT AGENCY FOR £3000
(Up to 50% loan available — more in exceptional case)

We are a consortium of agencies and this offer represents an unusual opportunity for a young person (preferably under 30) to run a business and earn up to 80% of the profits for very little capital. Those with experience in the agency field will know that this means very high earnings indeed.

We supply fully equipped prestigious offices, starting clients and applicants, advertising etc.

If you have excellent experience in a computer staff agency and can prove a record of success — Phone Colin Gray 499 5204 at Berkeley Street Employment Agencies 16, Berkeley Street, Mayfair, London W.1.

£800 to be won

PERSONAL COMPUTER WORLD

THE COMPETITIONS

As many people wrote in to say that there was not enough time to write up entries for the Competitions, the time has been extended. The Competitions remain open, and the closing date for entries is now the 31st of December 1978.

The categories are:

1) Homebrew Systems.
2) Software for games
3) School applications
4) Home applications.

The prize in each category is £200. The National Computing Centre has very generously donated the prize for the Home Applications category.

HOW TO ENTER

Entry is free. *Send for a copy of the Rules to:* Competitions, PCW, 62A Westbourne Grove, London W.2. Please enclose a stamped addressed envelope.

Advertisers

It pays to advertise *in* **PCW.**
★ It's the magazine people want to keep.
★ A highly intelligent and articulate readership with purchasing power to match.
★ Unique in Europe, from the quality of its content to its widespread distribution: on sale to the public at all good newsagents.
★ On sale at specialist shops.
★ Available at over thirty prestigious hotels.
★ Estimated sales of No. 2 – 27,000
★ Over 3,000 subscriptions.
It pays to advertise in PCW.
Contact the Advertising Department at 62A Westbourne Grove, London W2.
OR 'phone 01-727 8758.

Subscriptions

Having difficulty in ensuring your copy of *PCW*?
You can make sure of getting your copy by taking out a subscription. Rates (for twelve issues):
UK – £8
USA – $20
The Continent and elsewhere – £9.80
Subscriptions payable to Intra Press, 62A Westbourne Grove, London, W2.
Back numbers: Available to callers at the above address (newsagents, ground floor) or by post for 65p.

Apple Bally DEC Horizon Imsai MSI
APF Cromemco DG ICL IBM NCR
Altair CDC HP Intercolor Kim OSI

Get the most out of your computer!

creative computing

#1 in applications and software!

PET PolyMorphic SOL-20 SWTPC Tektronix Vector Graphic
TRS-80 Sorcerer Siemens Video Brain Wizard Xitan

No computer magazine gives you more applications than we do! Games. Puzzles. Sports simulations. CAI. Computer art. Artificial intelligence. Needlepoint. Music and speech synthesis. Investment analysis. You name it. We've got it. And that's just the beginning!

Whatever your access to computer power—home computer kit, mini, time-sharing terminal—*Creative Computing* is on your wavelength. Whatever your computer application—recreation, education, business, household management, even building control—*Creative Computing* speaks your language.

Read through pages of thoroughly documented programs with complete listings and sample runs. All made easy for you to use. Learn about everything from new software to microprocessors to new uses for home computers. And all in simple, understandable terms. And there's still more. *Creative Computing* discusses creative programming techniques like sort algorithms, shuffling and string manipulation to make your own programming easier and more efficient.

We can even save you time and money. Our extensive resource section is filled with all kinds of facts plus evaluations of hundreds of items. Including microcomputers, terminals, peripherals, software packages, periodicals, booklets and dealers. We also give you no-nonsense equipment profiles to help you decide which computer is best for you—before you spend money on one that isn't.

We've got fiction too. From the best authors in the field, like Asimov, Pohl and Clarke. Plus timely reviews of computer books, vendor manuals and government pamphlets. And so much more!

Subscribe today and receive your copy direct from America. Avoid disappointment in the event your shop runs out and save time and money by subscribing. Please allow 8 to 12 weeks for subscription to begin.

Back numbers (Volumes 3 and 4) and collected annual volumes (1 and 2) are available; also *Basic Computer Games; Microcomputer Edition,* the best-selling book of computer games in the world. See coupon for prices.

Available in selected computer shops and by post.

I want to get the most out of my computer.
Please enter my subscription to:

creative computing

• 12 issues per year

☐ 1-year £13 ☐ 3-year £36
☐ 2-year £25 ☐ Lifetime £225

Back numbers, £1.75 each

☐ Vol 3, No 1	☐ Vol 4, No 1
☐ Vol 3, No 2	☐ Vol 4, No 2
☐ Vol 3, No 3	☐ Vol 4, No 3
☐ Vol 3, No 4	☐ Vol 4, No 4
☐ Vol 3, No 5	☐ Vol 4, No 5
☐ Vol 3, No 6	☐ Vol 4, No 6

Best of Creative Computing books, £6.95 each
☐ Volume 1 ☐ Volume 2

Basic Computer Games book, £5.50 each

Amount enclosed _____

Name _____

Address _____

City _____

Postal Code _____

Send to:
Creative Computing, Attn: Hazel Gordon
Plot 23, Andrew Close, Stoke Golding,
Nuneaton CV 13 6EL

THE POWER OF POSITIVE TABLING

R.W. Davy

Introduction

The aim of this article is to show, by example, how another technique can be used by software designers to ease their task. I have deliberately chosen an example from process control because this is a major application area for microprocessors. Examples that appear to be totally different occur in many fields, such as stored program control of telephone exchanges, and it is part of the job of the software designer to detect the commonality of the problems and thereby to exploit available and tested techniques from superficially different areas. I had originally intended to use as my example the control of a central heating system, but a remark in the first issue of this magazine suggested that this has become a rather "old chestnut".

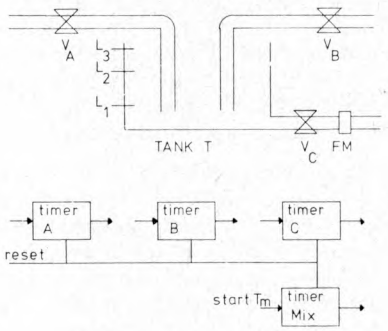


Figure 1

A Practical Example

The system to be controlled is shown in Figure 1. The system is to operate as follows:

Liquid A is to be admitted to the tank T through valve V_A until level L_1 is reached; this should not take more than T_A seconds. Liquid B is then to be admitted through valve V_B until level L_2 is reached; this step should not take more than T_B seconds. The batch is then to be left to mix for T_{mix} seconds. The tank is then drained through valve V_C; this step should not take more than T_C seconds. The process is then to be repeated. The contents of the tank should not reach level L_3; if the level does reach L_3, or if any other abnormal condition occurs an alarm is to be sounded. The plant is to be controlled every 100 milliseconds.

The three valves V_A, V_B and V_C are on/off valves, that is they are either fully open or fully shut. *The state of each valve can be determined by the processor.* The flow meter FM provides a logic true signal while fluid flows through it, and a logic false when there is no fluid flow. The level detectors L_1, L_2 and L_3 give a logic true when they are covered by fluid, and a logic false signal when they are not. In addition here are four timers; these give a logic true signal for a *set period of time after they are started,* and a logic false signal at all other times. The timers can also be stopped and reset simultaneously by applying a logic true pulse to their common reset input line.

Now that we have a specification for the operation of the system and for the instrumentation that is available we can start to develop the software that will, hopefully, control the system. One obvious approach is to draw a flow chart and Figure 2 gives the start of a typical first attempt.

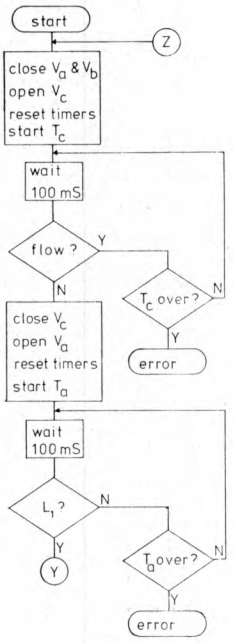


Figure 2

Although the program corresponding to the flow chart in Figure 2 would control the system adequately if there are no failures, it would *not detect* any failures if they did occur (and they will). This is obviously not within the specification given above and is therefore unacceptable. Consider a situation that could arise: level detector L_1 fails when valve V_A is opened. Under these conditions liquid A will fill the tank because there is no check on level detector L_2 while the program is expecting a signal from level detector L_1. Of course timer T_A should prevent this but it is human nature to back alarm limits off to reduce false alarms.

It is clear that a good many more checks need to be

incorporated, and the flow chart is going to grow large and cumbersome. This in turn will make it difficult to see if the flow chart is correct and complete.

There is an expression in use in certain programming circles, "to eyeball". If a program can be "eyeballed" then the viewer can see the structure almost directly he looks. "Goto" statements tend to prevent "eyeballing", that is to say that they obscure the structure of the program. The flow chart for this application is going to have a lot of (conditional) "goto's" in it, and it will probably be impossible to "eyeball" the flow chart.

Let us consider another approach, only using decision tables this time, instead of flow charts. The decision table to solve this problem is given in Table 1; it is a limited entry vertical rule decision table, or LEVRDT for short. In order to discuss decision tables we need a common nomenclature and this is provided by the skeletal LEVRDT in Table 2. You will see that a LEVRDT consists of four parts: *condition stubs, rules, action lists* and *action stubs.*

The term "limited entry" means that the rules consist of a limited number of types of elements, corresponding to "yes", "no" and "don't care". In Table 1, I have used "T" (for true), "F" (for false) and "–" (for don't care). It is perfectly feasible to have more types, for example an age in years, but the table then ceases to be of the limited entry type and becomes of the extended entry type.

The term "vertical rule" means that the rules (and action lists) run vertically and the stubs horizontally. This is most convenient typographically, and is generally easier to follow although this may be because of familiarity: the alternative, "horizontal rule", occurs less frequently.

Level 3 reached ?	F	F	F	F	F	F	F	F	F	-
" 2 "	-	F	F	F	F	F	T	T	T	-
" 1 "	T	-	F	F	F	T	T	T	T	-
Timer A running ?	F	F	F	T	-	F	F	F	F	-
" B "	F	F	F	F	F	T	-	F	F	-
" C "	T	T	-	F	F	F	F	F	F	-
" Mix "	F	F	F	F	F	F	F	T	F	-
Valve A open ?	F	F	F	T	T	F	F	F	F	-
" B "	F	F	F	F	F	T	T	F	F	-
" C "	T	T	T	F	F	F	F	F	F	-
Fluid flowing ?	T	T	F	F	F	F	F	F	F	-
Close valve A					1					
" " B							1			
" " C			1							
Open " A			2							
" " B					2					
" " C								1		
Reset timers			3		3		2			
Start A timer			4							
" B "					4					
" C "								2		
" Mix "							3			
Wait 100 mS	1	1	5	1	5	1	4	1	3	
Error										1
Repeat table	2	2	6	2	6	2	5	2	4	2

Table 1

condition stub	rule
action stub	action list

Table 2

Perhaps the easiest way of understanding decision tables is to see one work, and we can see the decision table in Table 1 work by going for a "walk through" it.

The first rule states that whatever the state of level detector L_2 and provided the level detector L_1, timer C, valve V_C and flow meter FM are true and all other conditions are false then the program should (1) wait 100 milliseconds and (2) repeat the table. If the first rule is not obeyed then the second rule should be applied.

The second rule states that whatever the state of level detector L_1 and provided timer C, valve V_C and flow meter FM are true all other conditions are false then (1) wait 100 milliseconds and (2) repeat the table. If the second rule is not obeyed then the third rule should be applied. The third rule states that whatever the state of timer C and provided only valve V_C is true and all other conditions are false then (1) close valve V_C, (2) open valve V_A, (3) reset the timers, (4) start timer A, (5) wait 100 milliseconds and (6) repeat the table. The execution of the other rules follows the same pattern.

I produced the table by the following method. I first assumed that a batch had just finished mixing and that it had started running out. Rules 1 and 2 correspond to the legal states that would be expected, and which require no action during the draining phase. Rule 3 corresponds to the state which indicates the tank is drained and the action list is the list of operations to be performed to initiate filling the tank with liquid A. Rule 4 corresponds to the state to be expected while liquid A is being run in to the tank, and rule 5 corresponds to the state which indicates that liquid A has reached level detector L_1. The action list for rule 5 is the list of operations to be performed to terminate the inflow of liquid A and to initiate the inflow of liquid B. The other rules and action lists were produced in the same way.

The last rule, rule 10, is an example of an "else" rule. The "else" rule, consisting as it does of "don't cares", will always be true although it is only reached if all other rules are not obeyed. It therefore guarantees that the table terminates properly; in this example all unexpected conditions will reach and obey the else rule and so cause the action "error" to be executed.

Having seen how decision tables can be constructed, and how they operate we should turn our attention to other matters. The first and perhaps most obvious matter we should consider is implementation. In order for decision tables to be practically useful as programming aids there must be a straight-forward way of converting them into computer languages. There are two main approaches, and they raise different questions, but before I go on to consider them we should first decide what is meant by "a good program".

Some properties of a "good" car
Satisfactory delivery
High acceleration and speed
Low fuel cost
No mechanical faults
Good documentation
Safe
Flexible

Table 3

Some properties of a "good" program
Satisfactory delivery
Fast execution
Low memory requirement
No bugs
Good documentation
Safe
Flexible

Table 4

The properties of a good program are rather similar to the properties of a good car, and it may help us to consider first what properties should be possessed by a car for it to be called "good". Table 3 is a list of some of the properties that I think are significant, and we notice straight away that *there are conflicts that will force the designer to make some compromise.* There is obviously a conflict between high speed and low fuel consumption; and although the conflict can be reduced by good aerodynamic and engine design it can never be completely

eliminated. These properties relate almost exactly to the properties I would expect a good program to exhibit, which I have listed in Table 4.

You may wonder what I mean by safe in the context of a program. Safety as I see it basically the art of *preventing small errors having greater repercussions than necessary.* I doubt that seat belts have a significant effect on the number of human motoring errors, but they do have a significant effect on injuries sustained in the ensuing accidents. Safety in a program is very similar, and it aims to reduce the effect of an internal or external error, should one occur. In our sample plant we should check the level detectors for self consistency, and do anything we can to *localise* errors.

Good documentation is desirable because bugs do occur from time to time, and they have to be fixed. Also good documentation aids flexibility, and flexibility enables us to obtain greater benefit from our investment. On the other hand the more jobs a program can do, the less well it can do any of them and this is another design compromise.

The aim of this digression into good programs has been twofold. Firstly, I shall need some of the ideas I have developed here later in this article. Secondly, because it is important to know what sort of features to look for when you are trying to assess software. If you are looking at Basic interpreters (and there are other ways of running Basic programs) you might want to consider some of the things I have listed in Table 4, such as how many bytes are allocated to each variable, is source test compression used, and a host of other questions. Cheddar cheese is not better than Stilton because it is heavier.

Let us now return to the question of implementation decision tables for microprocessors. There are, as I have mentioned earlier, two different approaches to implementing decision tables. They differ very considerably and produce very different sorts of program; by applying the relevant criteria of a good program we shall be able to assess the two alternatives. Let us first consider the conversion of decision tables to flow charts.

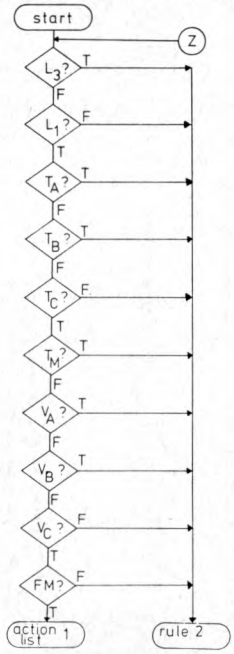


Figure 3

Conversion to Flow Charts

A decision table can easily be converted to a flow chart. We can adopt a mechanical procedure; take the first rule and for each significant condition (i.e. one that does not have '—' against it) perform the test indicated and if it fails jump to the next rule. Otherwise, deal with the next significant condition. If all the conditions obey the rule, then perform the action list. The application of this procedure to the decision table in Table 1 produces the flow chart in Figure 3; because the translation from decision table to flow chart is so mechanical it should be performed with few errors. So, we would expect to obtain a relaible program quickly.

A moment's reflection will tell us that this solution, simple as it is, is very inefficient in both execution time and program size. The inefficiency stems from the fact that several conditions will be tested more than once in arriving at one action list. In rule 2 we shall test level detector L_2 and if it is false we shall continue with rule 2; if level detector L_2 is true then the mechanical procedure will branch to start testing rule 3, although looking at the decision table it is obvious that if the level detector L_2 is true then rules 2, 3, 4, 5 and 6 cannot be obeyed.

A better approach for the implementation of our decision table might be to test "L_3" first and branch straight to "Error" if it is true. If "L_3" is not true then if we test "Timer Mix" we can, if it is true, branch to rule 8. We can repeat this process and with each decision produce two branches; the final result will be a tree, called a "decision tree". It is quite easy to pick out promising conditions to test first, but the chances of arriving at an optimal decision table (when doing the work by hand) in a reasonable time is slight.

A considerable amount of research has been done into the problem of finding a mechanism (usually called an algorithm) that will produce an optimal decision tree (and hence flow chart and program) for a given decision table. However, I do not propose to go into the algorithms so far found because there is an alternative method for implementing decision tables which produces programs that are well suited for microprocessor implementation.

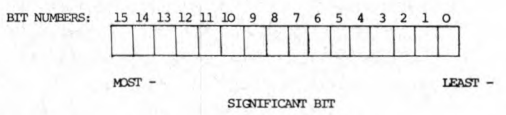


Figure 4

```
$0011. AND. $0101  =  $0001
$0011. OR . $0101  =  $0111
$0011. XOR. $0110  =  $0110
$01                =  $10
```

Figure 5

Conversion to Parallel Logic

A much less obvious method of implementing decision tables, but one that gives better results under certain quite common conditions, is *rule masking*. To go any further with this we must first develop some nomenclature in order to be able to discuss word logic.

A computer word is made up of a number of bits; the least significant bit is conventionally written on the right and referred to as bit 0, successive bits then have increasing bit numbers up to the most significant bit which in a 16 bit word is referred to as bit 15 (see Fig. 4).

Logical operations can be performed on words, in which case the logical operation is performed between bits in corresponding positions on a one to one basis. The common operations are AND, OR, EXCLUSIVE OR (XOR) and COMPLEMENT. The complement operation is the only unary operation, which is to say it has only one argument. The complement of X is represented by $\overline{X}$.

The contents of a computer word can be represented as a string of bits, and we prefix the string by $. The sample logical "sums" in Figure 5 give examples of all the logical operation we shall need. Most binary com-

puters can perform at least the logical operations indicated above on pairs of words in the same time (or faster) than they can add or subtract pairs of words. The significance of this from our point of view is that we can perform the same logical operation on 16 logic values (in a 16 bit processor) simultaneously. This is due to the technique for implementing decision tables by rule masking.

The first step is to produce the two auxiliary tables given in Table 5 and Table 6; they are produced by substitution in the rules of Table 1. Table 5 indicates the position of conditions which are significant i.e. there is a "1" in Table 5 wherever there is a "T" or "F" in Table 1. Table 6 shows the position of condition that must be true for the various rules to be obeyed, i.e. there is a "1" in Table 6 wherever there is a "T" in Table 1. These tables are called the significant condition table and true condition required table, respectively. *These tables have to be stored in memory as part of the program's fixed data area.*

1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	0
1	0	1	1	1	1	1	1	1	0
1	1	1	1	0	1	1	1	1	0
1	1	1	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	0

Table 5

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0
1	0	0	0	1	1	1	1	1	0
0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0
0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0
1	1	1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0

Table 6

Each time the decision table is executed we must produce a condition state word. Initially the word is $0. If the first condition is true, i.e. if level detector L_3 is true, then OR $1 into the condition state word. If the second condition is true, i.e. if the level detector L_2 is true, then OR $10 into the condition state word. If the third condition is true, i.e. if level detector L_1 is true then OR $100 into the condition state word, in which the most significant bit represents the state of conditions 11, the least significant bit represents the state of condition 1 and generally the nth bit represents the state of condition n+1.

The next step is to determine which action list to obey. To do this we copy the condition state word (since we do not want to rebuild it after testing each rule) into a *temporary* word W. We then AND the first column of Table 5 with W and compare the result with the first column of Table 6. If they are the same then the first rule is obeyed and we should execute the first action list, otherwise we repeat the process for each successive column in Tables 5 and 6. Notice that the "else rule" prevents an endless loop; whatever W may be, when it is ANDed with the last column of Table 5 the result must be $0 which is the same as the last column of Table 6.

A flow chart based on this rule masking approach is given as Figure 6. The reason I have labelled the end of the section which builds the condition status word is that there is another bonus in store for the rule masking approach. In most processers the condition state word would be read from an input port in a parallel form so that we would just input the external logic signals into a word, called CSW, and then start executing from label X.

Conclusion

By applying the suggested criteria of "goodness" to the programs produced from decision tables by use of the rule masking approach, it will be seen we can produce "good" programs from decision tables with little difficulty. Since decision tables themselves are an efficient method of software design for systems requiring a great deal of logic and relatively little computation, we can reasonably claim to have added another useful tool to our software toolbox.

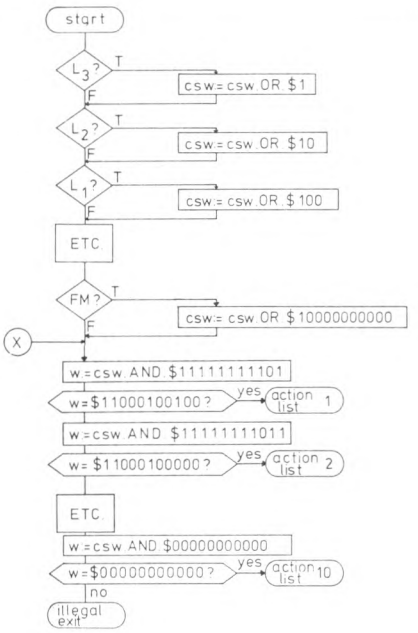


Figure 6

Numerous problems have been glossed over; this is inevitable in a short article on a subject that has been the subject of many papers, monographs and books. As an example the single statement "wait 100mS" conceals the problem of *what to do with the free time;* the solution is a real time scheduler which in turn is part of a real time operating system. I hope in future articles to explore these and other matters. In the mean time the reader who would like to know more about decision tables is recommended to look at "Programs from Decision Tables" by Humby and Macdonald/American Elsevier Computer Monograph series (in which it is number 19).



"This is the computer, that's the soft drinks machine!"

For Sale:

KIM 1 Microprocessor Board (ready
KIM 4 Motherboard assembled)
CRT Card – Thomson-SE-KEX 68364.1.0
ASCII Keyboard, (free-standing ex VDU keyboard housed in attractive two tone case; keyboard roll over).

Original cost £456.00 (incl VAT) will sell collectively at £325 to obtain quick sale.

Tel: Cheltenham 29135 Monday – Thursday 8.30 am to 5.30 pm further details.

GODBOUT
Computer Products

Altair/Imsai/Cromemco/Polymorphic Etc.
S–100 Bus Computer Compatible Products

LTT ELECTRONICS has one of the largest stocks of Godbout computer products in the U.K. Only by bulk purchasing and minimising sales overheads can we afford to sell at what amounts to trade prices.

For example:	**Kit**	**Ass.**
Econoram II, 450ns 8K bytes static memory board:	£ 85	£ 99
Econoram IV, 250ns 16K bytes static memory board:	£175	£195
Econoram VII, 250ns 24K bytes static memory board:	£255	£275

All prices include postage and insurance (overseas add £10) – Just add 8% VAT
Send 25p (overseas: £1) for full details.
MAIL ORDER ONLY

LTT ELECTRONICS Department 16
37 Orlando Road London S.W.4.

STAR TREK

Now available on cassette a version of startrek utilising the graphic capabilities of the Commodore PET. Destroy the klingon invasion force before they destroy the Enterprise. Complete operating instructions (£4.50).

Send cheque, postal orders payable to M. Green 57 Ross Road, South Norwood London SE25 6SB.

NASCOM 1 READY BUILT MICRO-COMPUTER FOR SALE

Full 48 key keyboard, Z80 CPU, 16 x 48 Character Display, Memory can be expanded to full 60K.

All fitted in a nice case, also included a cassette recorder and 12" TV also software, six games tapes. Only £320 ono. **Tel: East Horsley 3709.**



PLAN FOR A PLOTTER

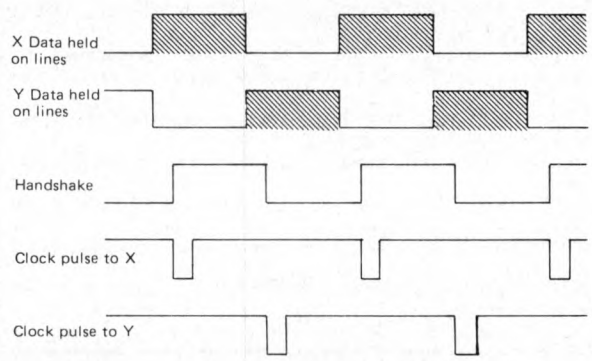
R.I. Elliott

The problems involved in interfacing the Pet 2001 to an X–Y plotter are complicated by the fact that it does not have a standard bus structure that is accessible. The I/C.s available consist of one user port of eight data input or output lines and two handshake lines, one of which is only an input line. The other port is configured as an IEEE bus. It was decided not to use this port as it was to be used later as a hard copy output.

As I needed to output both X and Y data I decided to multiplex the data in the software and to provide a latch in the external circuitry.

The latch consists of two CD4042 integrated circuits to handle the eight X data lines and two similar circuits to deal with the Y lines. The problem now arising was how to refresh the X and Y latches at different times using only one available handshake line. This was achieved using CD4047 monostables which were set to trigger on the positive or negative going edge of the handshake line.

IC1 and IC2 are the eight bit (2 x 4 bit) latches for the X data and IC1a and IC2a are the latches for the Y lines. These latches have clock and polarity inputs. The state of the polarity input determines whether the chip latches on the negative or positive going edge of the handshake pulse. It was decided to use the positive edge to store data and the negative edge to refresh the latches. Therefore the polarity pins of the ICs were connected to low.



DAMS Ltd

DAMS LTD. (COMPUTER DIVISION)
30-36 DALE STREET LIVERPOOL L2 5SF.
Telephone 051 227 3301.

COMMODORE PET

Commodore Pet	£ 643.52
Second Cassette Unit	£ 55.00
Joystick (with software)	£ 25.00
Apple II (4k RAM)	£ 995.00
With 16K RAM	£1250.00
With 48K RAM	£1900.00
NASCOM I Kit	£ 197.50
Power Supply	£ 24.50
Socket Pack	£ 4.90
Memory Expansion Board	£ 35.00
Buffered Bus Board	£ 25.00
NASCOM I/O Board	£ 35.00
KIM I Microprocessor	£ 149.00
KIM 3B 8k Memory Expansion	£ 179.00
KIM 4 Motherboard	£ 89.00
KIM 5 Resident Assembler	£ 129.00
KIM 6 Prototyping Board	£ 39.00

SCIENCE OF CAMBRIDGE

Science of Cambridge Mk 14	£ 39.00
Monitel Telephone Charge Clock	£ 26.00
International Monitel	£ 36.00
Checker Challenger	£ 92.00
Chess Challenger	£ 129.00
Gammon Master II	£ 157.40
Mind Reader	£ 299.00
Apple II Disc	£ 423.00
Texas Programmer	£ 46.25
Sinclair Digital Multimeter	£ 27.00

Add VAT at 8% and 5op p.p. on each order.
Cash with Order. Trade enquiries and callers welcome.
Cheques to DAMS LTD.

DAMS

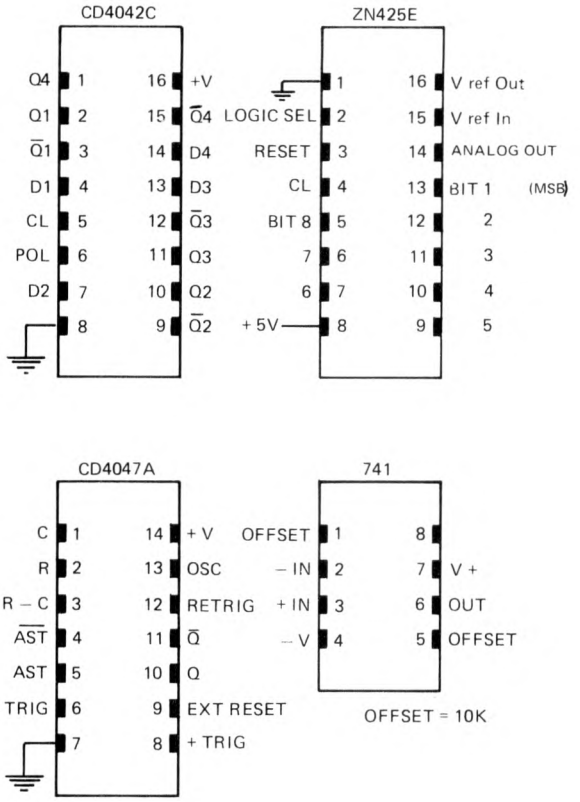
Dams stock Books, Documentation, CMB and PETSOFT Software and a comprehensive range of TTL and CMOS devices. *Send SAE for full lists.* Here are a few example of *our* software on PET compatible tapes at *£4.00 each.*

GAMES

SWAT swat an exclusive fly TORMENTING.
RACE steer a ball round tortuous route at various speeds.
DEFLECTION very enjoyable, skillful ball angles game.
TREASURE DIVE recover treasure in the face of hostile harpoons.
DUCKSHOOT variable speed flapping ducks, including retreiver.
VARIABLE IQ NIM the machine plays at your level automatically.
BRAIN STRETCHER similar to 'mastermind' or 'bagels'.
MUSIC plays recognisable tunes (circuit for SPKR system included).
LIFE fast machine code version of this enthralling cult-game.

OPERATIONAL

LINEAR REGRESSION with error routines.
15 DEEP SIMULTANEOUS EQUATION SOLVER
ALGERNON XYZ evaluates any function for up to 3 variables.
POLAR PLOT plots polar graphs of any function.
DISTORT (specify cartesian or polar) axial rotation, scaling, origin displ, relative axial tilt.
ELEQTRON computation of Inductance, Capacitance and Resistance. Inductance winding parameters, 'Q' and Dielectric loss.
SPECTRET DQ Ligand Field Theory, evaluation of DQ for spectral parameters of d^5 ions.
DOUBLE DENSITY machine code high density plotting subroutine.
LINE RENUMBER machine code renumbering of program lines.
MACHINE CODE HANDLER loading and inspecting of *all* locations
ACCOUNT PREVIOUS & PRESENT invoice record with running balance.
QRA CODE LOCATOR (for radio Hams.)

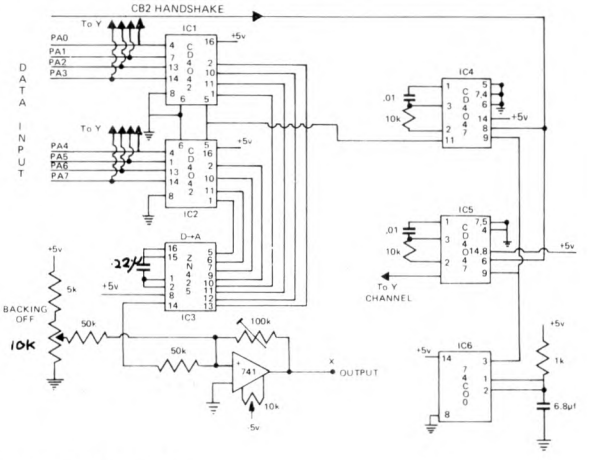


PIN CONFIGURATIONS

The data inputs from the Pet user port were fed to the D inputs of the latches and the outputs taken from the Q pins to the inputs of the ZN425 D–A converter IC3 and IC3a. The analog output of the ZN425 is taken to a 741 op-amp which is used to buffer the output. The 741 op-amp which is used to buffer the output. The 741 is also used to set the gain required for the X–Y plotter in use and to level shift the trace to the correct position on the X and Y axis.

The output lines of the Pet are inverted i.e. 255 entered in the output register gives a 0 on the output lines. A saving in components can be made by reinverting the signal at the op-amp stage, this will give a level change from –fsd. to 0v thus a backing off voltage is needed at the op-amp to shift the –fsd to 0v and the 0v to +fsd.

The circuitry that enables the X latches and the Y latches at the appropriate times consists of IC4, IC5 and IC6. IC6 is used to ensure that IC4 and IC5 power up in the correct state. IC4 and 5 are CD4047A monostables with an output pulse width determined by the 0.02uF capacitor and the K resistors. IC4 is arranged to



CIRCUIT DIAGRAM

PET 2001

TRS 80 Level 2

APPLE II

"business applications software"

WE CHOOSE THE HARDWARE TO SUIT YOUR REQUIREMENTS

WE PROVIDE THE SOFTWARE THAT WILL SUIT YOUR NEEDS **EXACTLY**

WE **GUARANTEE** THAT, IF YOU ARE NOT **COMPLETELY SATISFIED,** EITHER WITH THE EQUIPMENT OR THE SOFTWARE, IT WILL NOT COST YOU A PENNY

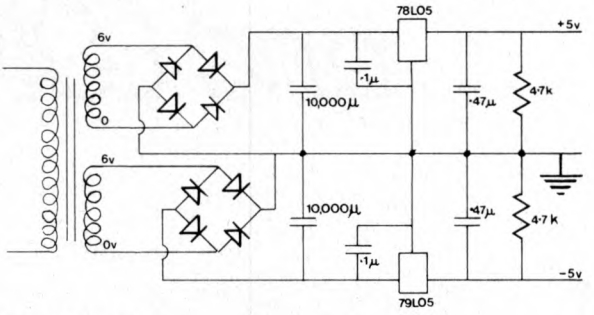
PRICES START – Including Software FROM AS LOW AS £2400.

Please reply to:-

Mr. D. C. JAMES,
EDINSMITH COMPUTER SERVICES Ltd.,
11, Stoke Hill,
Stoke Bishop,
BRISTOL, BS9 1JL.

trigger on the positive going edge of the handshake signal. This is achieved by connecting pins 4, 5 & 6 to earth and applying the handshake to pin eight. IC5 triggers on the negative going edge by tying pins 4 & 5 to earth and pin 3 to +5v the handshake is when applied to pin 6. The Q outputs are used from these ICs, this ensures that the output is always high except for the delay time after triggering. At this time it goes low and refreshes the latches.

The power supply is fairly simple, using R.S. Components parts. The supply has to deliver about 50mA therefore with 78LO5 and 79LO5 are well within their capabilities.



POWER SUPPLY

```
30 POKE59459,255
40 POKE59468,PEEK(59468)AND31OR192
50 POKE,X
60 POKE59468,PEEK(59468)OR224
70 POKE59457,Y
80 REM LINE 30 SETS PORT TO OUTPUT
90 REM LINE 40 SETS HANDSHAKE LINE LO
100 REM LINE 50 OUTPUTS X
110 REM LINE 60 SETS HANDSHAKE LINE HI
120 REM LINE 70 OUTPUTS Y
```

Components List

Semiconductors

IC1 & 1a	CD4042A
IC2 & 2a	CD4042A
IC3 & 3a	ZN425
IC4	CD4047A
IC5	CD4047A
IC6	74COO
Op-amps	741 x 2

Capacitors

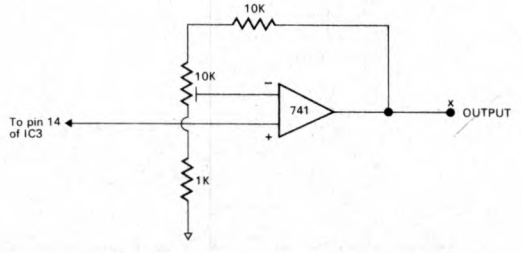
0.22uF x 2
0.01uF x 2
6.8uF x 1

Resistors

5K x 1
50K x 2
10K x 2
1K x 1
10K preset x 2
100K preset x 2

Power Supply

6–0, 6–0 transformer (100mA)
bridge rectifiers
10,000uF capacitors x 2 1000 uF will work and save space)
79L05 regulator
78L05 regulator
0.1uF x 2
0.47uF x 2
4.7K resistor x 2



BENCHMARKS AGAIN and CONNECTIONS

John Coll

Below are some more benchmark results. I am still receiving figures from all over the place, as far afield as Norway and the World Health Organisation in Switzerland! The listing of the tests was published in Personal Computer World (Vol.1 No.1) or can be obtained from me.

RS232/V24 Connections

Connections between a computer and its peripherals are often made by a standard 25 way plug and socket using signals conforming to the RS232 or V24 standards which are, effectively, interchangeable.

The MSI computer system have modified the standard connections to permit automatic Baud rate selection by the plug. This means that each peripheral's plug is wired to tell the processor at which rate it operates so no manual switching is required. This seems a sensible decision.

The standard "Cannon 25 way plug" is available from **RS Components** under part number 466-191. The connections most frequently required are as follows:

Pin 2 RS232 Transmit data
3 RS232 Received data
5 Clear to send
7 RS232 Ground
10 110 baud clock source
11 150 baud clock source
12 300 baud clock source
13 600 baud clock source
15 1200 baud clock source
23 clock input

Other pins ARE allocated, for example

Pin 17 IRQ enable jumper
18 IRQ enable jumper
21 logic 0 (same as pin 7)
22 -12 volts
25 MPU reset

To wire a plug to select a particular Baud rate connect pin 23 to the appropriate clock source pin. This information was kindly supplied by Ian Parsons of **Strumech Engineering Ltd.**, Brownhills, Walsall.

Benchmarks

Hardware	Software	yr	f	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	BM7&8
Wang 2200VP	Resident			0.3	0.7	2.2	2.3	2.5	4.1	5.3	1.0	6.3
IBM 370/115	CCUL Lisbon R1.2			1.3	1.4	2.3	1.9	2.1	3.1	4.6	1.9	6.5
Eclipse	RDOS			0.5	1.8	4.3	5.0	5.1	9.0	13.0	1.5	14.5
HP 9825A	Resident			0.7	1.8	4.0	4.4	4.7	9.0	12.1	3.7	15.8
HP 9845	Firmware 2		12MHz	1.1	1.5	3.4	3.6	4.3	10.2	14.2	3.6	18.4
Olivetti P6060	Resident			2	3	4	18	17	18	19	-	-
NOVA 1220	Time shared V 091-26-04	71		2.3	4.5	8.4	10.5	11.0	16.9	23.8	2.7	26.5
NOVA 1210	Stand Alone Sys. V9.02			1.0	3.0	7.7	9.1	9.6	16.5	24.0	2.6	26.6
PDP8A Classic	OS/8 R3			2.7	2.5	5.5	6.1	6.7	19.2	24.2	3.0	27.2
Apple	Apple II Integer B			1.5	3.2	7.3	7.2	8.9	18.6	28.2	-	-
PDP8E	Educomp V3.4			1.4	4.4	8.5	11.8	9.3	15.8	27.8	3.7	31.5
NOVA 1220	Multi-user Ex B SOS R9			1.8	4.0	10.0	11.6	12.4	20.8	29.8	3.8	33.6
T.I. 9900-10	Pre-release	77		3.2	2.4	4.5	5.5	6.1	25.0	29.0	-	-
R.M. 380Z	TDL Zapple 8k V1.3	77	4MHz	1.4	6.5	13.2	13.9	15.0	22.3	31.6	6.2	37.8
NOVA 2-10	Ex B R4.0 RDOS R3.03			1	3	8	9	9	16	25	16	41
NOVA 1210	4k Single user	73		0.6	1.8	5.8	7.3	7.6	12.5	18.2	32.2	50.4
ALPHA LSI 2-20	CAI BASIC-2			2.2	7.0	14.5	15.0	16.0	29.5	48.0	3.7	51.7
Z80	TDL 8k V1.3		2.5MHz	1.8	8.5	18.4	19.5	21.6	31.6	44.7	8.9	53.6
SWTPC MP68	Altair 680b V1.0 R3.2	76	898kHz	1.4	9.0	16.8	18.1	20.0	31.0	45.2	8.5	53.7
SWTPC MP68	3k integer		898kHz	1.6	8.7	15.6	16.4	21.6	31.4	46.5	-	-
PDP8M	Edusystem 10			3.0	4.6	14.6	15.0	15.4	34.0	47.1	7.0	54.1
R.M. 380Z	Palo Alto Tiny MBI.8	77	4MHz	0.9	8.6	13.6	17.2	21.2	-	-	-	-
Apple	Applesoft Ex B	77		1.3	8.5	16.0	17.8	19.1	28.6	44.8	10.7	55.5
PET 2001	Resident	77		1.7	9.9	18.4	20.4	21.7	32.5	50.9	12.3	63.2
PDP8L	8k			4.0	6.8	17.0	20.2	21.0	38.8	57.0	10.8	67.8
Altair 8800b	Altair Extended V4.0	77	2MHz	1.9	7.5	20.6	20.9	22.1	37.0	58.5	9.9	68.4
PDP8L	Edusystem 10 4k			4.8	7.0	16.8	20.2	20.5	44.8	61.5	9.5	71
Wang 2200T	Resident			4.4	9.5	24.4	23.1	25.9	49.3	74.9	13.6	88.5
Altair 8800b	Northstar V6 R2	77	2MHz	2.4	9.0	16.8	31.3	33.4	50.1	72.5	22.0	94.5
Altair 680b	Altair R1.1	76	500kHz	2.6	16.4	30.9	33.7	36.6	56.0	81.9	15.0	96.9
Tektronix 4051	Resident			4.7	14.2	33.3	36.1	40.8	69.0	103.9	14.7	118.6
PDP8L	DEC 4k CINET			4.8	29.0	45.8	55.2	60.0	73.0	133.6	12.0	145.6
HP 9830A & B	Resident		8MHz	4.4	14.6	35.6	38.1	40.5	74.3	128.4	19.4	147.8
IBM 5100	Resident			4.5	21.1	57.4	54.5	59.0	88.2	174.9	26.7	201.6
PDP8E	Edusystem 20 non alpha-num S.U.			8.5	18.0	52.3	52.4	65.4	65.4	203.7	12.4	216.1
PDP8L	Edusystem 20 12k			12.5	39.2	74.6	73.8	91.0	91.0	288.5	16.5	305
PDP8E	Edusystem 20 config for 1 user			12.5	47.3	86.0	86.1	96.4	96.4	333.6	18.3	351.9
SWTPC MP68	8k V2.0	77	898kHz	15.6	25.4	96.9	105.9	109.8	174.5	205.1	172.0	377.1
Mycron 8080A	Tiny B	75	2MHz	12.1	20.9	57.6	58.4	101	204	306	80	386
Elliot 803	8k V1	69		9	12	29	32	48	41	372	38	410

Hop on a Nasbus to 32K of memory now

The Nascom – 1 is designed with expansion in mind. This is made possible by using the best products available. The Z80 microprocessor incorporated in the basic system is so powerful it can support 64K bytes of memory and 256 ports. To utilize this capability, we have designed the buffered 77 – way Nasbus.

With this arrangement, the way is clear for considerable expansion, starting with our new memory expansion board. It has 16 memory sockets and two EPROM sockets. Therefore, you can fill it with 4K dynamic RAM up to a maximum 8K or with 16K dynamic RAM up to a maximum of 32K. A 2K Tiny BASIC in EPROM has been developed for the board.

To go with the board, we have produced a very flexible I.O board with three PIOs each giving two, 8bit ports, plus a UART for serial interface.

As you start building up your Nascom system you will need a convenient means of storing boards. Our new, custom-designed, Vero frame will allow for a Nascom – 1 to link through a buffer board to a 77-way Motherboard. There is then the option of eight or more expansion boards. To power this capability there is a new 8.5 amp power supply especially designed for the frame.

No other system offers so much at such a low cost. And it all starts with the basic Nascom – 1 kit which for just £197.50 offers an intelligently usable system with video and cassette interface, a full alpha-numeric keyboard and a mighty CPU chip. So if you want the best – make it a Nascom system.

Nascom-1 Kit still only £197.50 +VAT



Nascom Expansion – Prices

Buffer Board Kit	£25.00
4K RAM Board	£70.00
16K RAM Board	£179.00
4K RAM (4027)	£35.00
16K RAM (4116)	£106.00
Tiny BASIC	£25.00
2.2amp Power Supply	£24.50

Prices exclude VAT

UK National Distributors

Barrow-in-Furness
Camera Centre
Tel: 0229-20473

Torquay
CC Electronics
Tel: 0803-22699

Egham & Manchester
Electrovalue
Tel: 07843-3603

Glenfield, Leicester
Eley Electronics
Tel: 0533-8771522

London W2
Henrys Radio
Tel: 01-723 1008

Oldham, Lancs
Lock Distribution
Tel: 061-652 0431

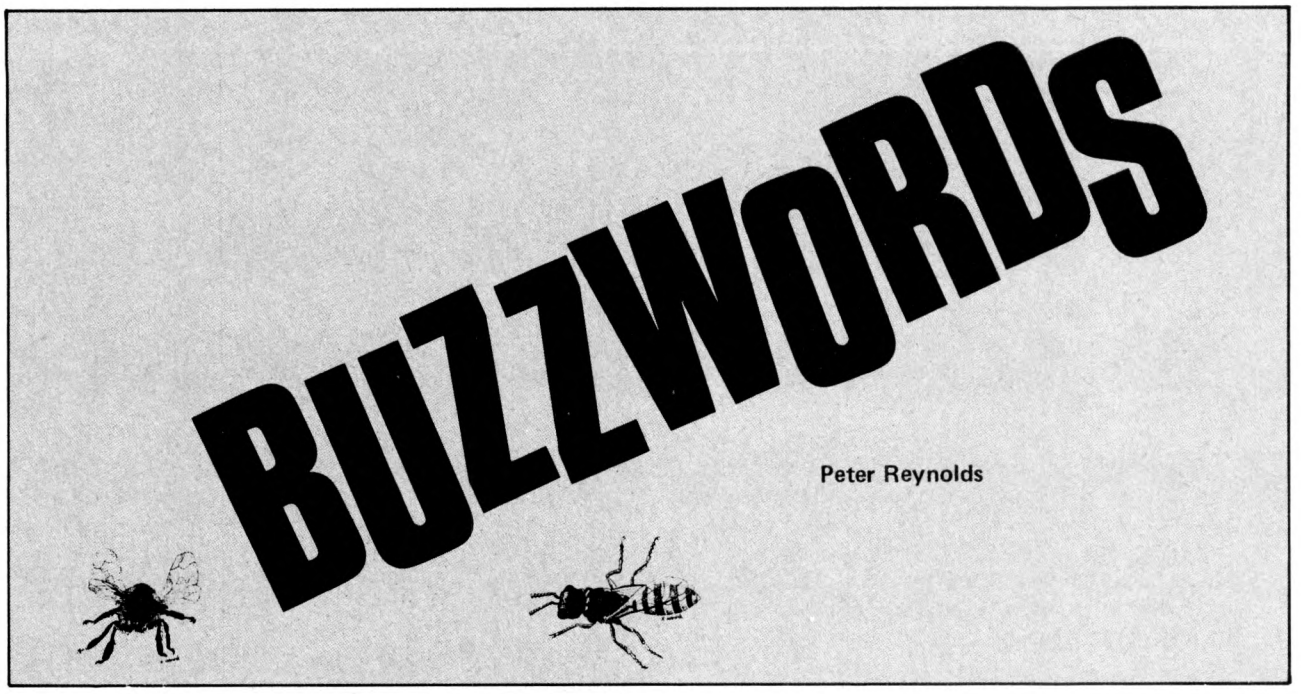
Chesham, Bucks
Lynx Electronics
Tel: 02405-75151

Liverpool L2
Microdigital
Tel: 051-2360 0707

New Barnet, Herts
Comp Components
Tel: 01-441 2922

Nascom Microcomputers
92 Broad Street, Chesham, Bucks.
Tel: (02405) 75151

PCW1178



Capacitance – a measure of the effective value of a *capacitor,* normally measured in microfarads (μF or sometimes mfd), nanofarads (nF, one thousandth of a microfarad) or picofarads (pF or 'puff', one millionth of a microfarad). Capacitance exists between conducting surfaces separated by an insulator and is a function of the areas of the two surfaces or plates, the distance separating them and the insulating material or dielectric.

Capacitor – a widely used electronic device comprising two conducting plates narrowly separated by some insulating or dielectric material and with the properties of storing electrical energy and delaying a change of voltage.

Card
(1) Circuit card. A sheet of plastic or similar insulating material on which electrical and electronic components are mounted and interconnected to make a functional circuit.
(2) Punched card. A piece of stationery, typically measuring 187 by 82mm, used to record data for subsequent reading by machine as a means of input to a computer. A popular size of punched card can record up to 80 alpha-numeric characters.

Card Image. A representation of the contents of a punched card held ready for data processing. The card image is normally picked up by a card reader and held in the *buffer* attached thereto.

Card punch. A device for recording machine – readable data by punching holes in a card. The holes will conform to a coded format, such as the *Hollerith Code,* qv.

Card Reader. A device for reading (usually by photo-optical means) data represented by means of holes punched in cards such as those made by I.C.L. or by I.B.M. – for transfer to the input area of a computer.

Carriage Return. The typewriter key (or its equivalent ASCII code (101 1000)) which causes the next printing position to be at the beginning of the line. In computing the CR signal usually means that a line of data or instruction has been completed and should be transferred from the buffer in which it has been built up to the central processor, for whatever action is appropriate.

Cathode. The negative electrode of a pair, the positive one being called *anode.*

Cathode Ray Tube. A cathode ray tube is the glass screen component on which a television picture or radar scan may be seen. A CRT terminal is a peripheral device on which computer information may be shown much as a television picture is shown. The display may take the form of a line diagram or a picture or, very commonly, a block of letters and figures. Since characters are generated electronically it is possible to have a large selection available and to include graphing shapes as well as representations of 'non-printing' characters such as tabulator or bell commands.

Ceefax. The British Broadcasting Corporation's name for its television screen message service, feeding a computer-like display to the (modified) domestic receiver. Comparable developments by the Independent Broadcasting Authority and the Post Office are respectively called *Oracle* and *Prestel* (formerly *Viewdata*).

Central Processor. The main part of a computer, containing immediate access storage, arithmetic and logical units and special register groups. In effect, it is those parts of a computer other than input, output and peripheral devices and, in some cases, also excluding immediate access storage areas which may be provided in separate modules. Abbreviated as CPU.

Chad. The circular piece of paper or card removed when a hole is punched.

Chain (verb). Two or more programs may be chained together so that they run as a single entity without attention. For example Program 1 may check the validity and completeness of data to be processed and end with an instruction to 'chain to program 2' which does the actual computing and 'chains to program 3' which updates files and produces reports.

The point of chaining is to allow a computer to run a number of smaller programs, one after the other, when a single larger program would be beyond its capacity. This subdivision may simplify the task of the programmer as well.

Chain Printer. A form of line printer in which the type faces are carried on an endless chain which turns continuously. As the appropriate part of the chain passes in front of hammers, it is struck and so printed "on the fly." The letters comprising a whole line of text are struck so rapidly that it looks as if the whole line is printed simultaneously.

Channel. A path along which signals can be sent – for example, a telephone wire.

Character. Any of the *alphanumeric* and related symbols (punctuation marks, etc.) as found, for instance on a typewriter keyboard. A character must be represented within a computer by a pattern of binary bits.

Character recognition. Automatic reading by a computer device of characters which can also be understood by people. This is at present practical only with type faces conforming to a rigid format such as the OCR B alphabet used in some computer-produced Giro transfer forms or the MICR imprint on bank cheques.

Charge-Coupled Device. A type of memory, based on what are effectively chains of very small capacitors, with both digital and analogue qualities (the size of charge can be varied and read and not just the presence or absence of a charge). Data may be 'rippled' through the elements of a CCD, giving a precise period of delay or it may be retained statically for a relatively long period, though eventually subject to leakage loss.

Chassis. The rigid frame, usually of aluminium or steel, on which the majority of heavier components are fitted in an electronic device. A chassis may have circuit boards attached to it and the whole is usually contained in a metal or plastic case.

Check digit. An extra digit, designed to reduce the likelihood of a number being misread when it is entered in a computer. The check digit, which comes at the end of the true number, might be the remainder after dividing by nine. Then 3638 should be expanded for computer entry to 36382. If this inadvertently entered as 36882 a computer check routine should detect an error as the check digit for 3688 would be 7.

Chip. An integrated circuit (more strictly, the silicon chip on which a monolithic circuit may be built).

Circuit. A network of electrical paths whereby a current may flow from, and return to, its source.

Clear. To change all the information held in a computer register or buffer to zero in preparation for the receipt of new data.

Clock. The master oscillator which generates the timing impulses for a computer's central processor. See *timing logic,* PCW Volume 1 Number 2, page 17.

Clock Frequency. The basic speed at which a computer operates — expressed in terms of the frequency (for example, one megaHertz) at which the master oscillator operates and at which it generates the pulses which are the basis of all data transmissions within the computer system.

Clock, real time. A computer's master oscillator (see *clock*) which can also relate to calendar time and use this information as a computing input — for instance to calculate the cost of telephone calls according to the time of day and day of the week or year.

Clock Speed. The counterpart of *clock frequency,* where the duration of each cycle is stated (e.g. 250 nanoseconds) instead of the number of cycles occurring in each second (in this case 4 million or a frequency of 4 MHz).

Clock Track. A pattern of digits recorded on one of the tracks of a magnetic disc as a check on its speed of rotation and to permit identification of the data areas passing under read/write heads at any instant.

Close (a file). Terminate (by subroutine) a program's reading or writing access to a data file.

Closed Loop.
(1) A sequence of program instructions arranged to operate in a repetitive cycle which can end only with manual intervention — for example, when an operator presses a switch. A closed loop is sometimes written into a computer program to come into effect in the event of some error condition, for example, if data read in fails a validity test, and it will normally be arranged to set up an indicator such as a particular pattern of lights on the operator's console. Such a loop may require only a single instruction in an assembly language, for example, 'Jump Display 07777' which will cause twelve consecutive lights on the console to come on, representing 111 (the octal 7) four times.
(2) In a process control system a closed loop may signify a feedback arrangement (as in a radio receiver) where a measure of results can be used to make automatic adjustment of operating conditions earlier in the system. The governor of a steam engine provides a mechanical analogy; when the flywheel turns too fast it restricts the flow of steam from the boiler.

Closed Subroutine. A sequence of instructions which may be used more than once in a program and which provides a variable indication where control is to be taken thereafter. The first instruction in the sequence might note the address of the program instruction last obeyed, advance this number by one position and store it in location Z. The last instruction in the subroutine will read 'Jump to the instruction whose address is now stored in location Z' (an example of an indirect address). Without such an arrangement the subroutine would require to be re-written in full each time it is to be used, as an *open* subroutine. In BASIC programming a closed subroutine may be entered by a number of GOTO instructions at different points in the main program while a single RETURN instruction at the end of the subroutine will always find the place of last branching from the main program.

CMC-7. A standard font of characters for magnetic ink imprint suitable for automatic computer input by MICR devices.

Co-ax. Abbreviation of *coaxial,* where one conductor is in the form of a pipe and a second is a wire, held in the centre of the pipe by insulating material. Coaxial cables, plugs and sockets are familiar for connections with television aerials and they can be equally important where high frequency signals are used in computing.

Cobol. Common Business Oriented Language (acronym). An international standard program language intended for commercial use.

Code.
(1) Verb. To write program instructions in the specific terms required for the chosen programming language.
(2) Noun. A representation of data or instructions in machine-acceptable form, normally at low level like *machine code* or *ASCII* in contrast to a high level language.

Coder. Person who writes a program, following general instructions prepared by, say, a systems analyst.

Coding Sheet. A form prepared for programming instructions to be written on it, partly to ensure that no part of the instructions gets overlooked and partly to provide a uniform document from which keyboard operators can work at highest efficiency. Synonymous with 'program sheet' and 'coding form'.

Colour code. A convention for marking the value of resistors and capacitors by coloured bands on the bodies of these devices.

Cold Fault. A fault condition in a computer that becomes apparent as soon as the device is switched on, as opposed to a fault arising during running.

Collate (physically). To merge two or more sets of data to produce a single set in some desired sequence.

Collate (logically). To compare two sets of data in different areas of computer storage in order to extract whatever parts of those two records happen to coincide. This technique may be used to extract a particular piece of information which cannot be located by direct addressing (because it is a part of a larger unit).

Collating Constant. A number, or pattern of binary digits, used for logical collating as a form of *mask* for extracting a particular pattern of information which may present in the date subjected to this test.

Collator. A punched card machine which interleaves cards from two packs when information punched in one or more fields coincides.

Collector. One of the elements in a transistor, the others usually being *emitter* and *base.*

COM. Computer Output on Microfilm (acronym). When it proves necessary to store a great amount of processed details off-line, yet have it accessible for reference on demand this may be achieved by a device which converts the lengthy computer print-out directly to compact pieces of positive film. Special reader devices provide the magnification necessary to make a selected part of the film legible on a screen or to take a full-size photocopy.

Command. A computer control signal, for example, an instruction to shift information from location X to location Y in the computer store.

Common. The designation, at the beginning of a series of programs that will be chained together, of the names of files or variables that will be used in more than one of those programs (e.g. in FORTRAN).

Compare. A computer function which examines one number in relation to another and commonly sets switches according to whether the first is greater than, equal to or less than, the other. It is usual to follow the compare instruction with a conditional jump, for example, part of a budgetary control program might read:—

```
START   1. Read an item of expenditure.
        2. Compare expenditure with standard.
        3. If equal jump back to START.
        4. If greater than standard jump to HIGH.
        5. Print out item on list of underspendings.
        6. Jump back to START.
HIGH    7. Transfer to tape record of overspendings for
           subsequent analysis.
        8. Jump back to START.
```

Compatibility. The characteristic of computer hardware and software, based on conformity with a growing number of international standards which allows interchange and interconnection.

Compile. To prepare an *object program* in *machine language* from a *source program* written in *mnemonic code.*

Compiler. The computer program (manufacturers' software) whereby a computer can convert a source program into an object program expressed in machine language which the computer can

recognise. In addition to the translation function of assembly – changing each mnemonic instruction into a machine code instruction on a one for one basis – a compiler may generate additional program instructions indicated by more powerful mnemonic instructions; a complete subroutine may be initiated by a single source program instruction. In addition, as with the assembly routine, the compiler will allocate storage areas as required in place of the symbolic addresses used in the source program.

Computer
(1) Strictly, any device capable of solving problems by accepting data, performing prescribed operations on it and supplying the results of these operations. Every desk calculator is a computer in this sense.
(2) In common usage, a data processing machine which operates by its own internally stored program.

Concurrent Working. A system whereby peripheral devices in a computer configuration can operate simultaneously because each occupies the central processor for only a part of its operation cycle. Thus a cassette tape reader requires only a small time to sense and signal the data represented by one character or word and a relatively much larger time to move the tape into position for reading the next work. During this interval data may be transferred from the central processor to, for example, a printer. The actual printing will occur while further words are being sensed. Concurrent working is to be distinguished from *multi-programming.*

Conditional. Subject to the result of a comparison made during a computation.

Conditional Transfer/Jump. A computer instruction from which the computer will either continue in normal program sequence of further instructions or will transfer to another stated instruction, according to the result of a prescribed test. For example, each cycle of a program loop may deduct one from a number stored in a register and at the end of the cycle there may be a comparison of the number remaining in that counter with a predetermined number, for example, zero. When the number is reduced to zero the condition is met which permits transfer to an another part of the program, so that the cyclical operation on the loop is ended.

Configuration. The particular assembly of computer hardware selected by the owner, typically comprising processor, additional RAM and/or ROM, disc and/or cassette controllers and drives, keyboard and display device but with great scope for possible variation.

Connector. A flow chart symbol used to indicate two or more connected points; they will be marked with a common number or letter. See PCW Volume 1, Number 1, page 39.

Console. The piece of equipment provided with most computers from which all the apparatus can be controlled, and in which any requirement for attention is normally made evident. Apart from a few switches controlling the main items in a computer configuration, it used to be common practice to provide banks of additional switches and lights from which could be read the current state of certain parts of a computer's internal store. The current trend is to provide a typewriter as a means of communication between operator and the machine. A notification when a program is for some reason unacceptable, or that a piece of work has been finished, or that some device needs new supplies of paper, is now given on the typewriter.

Constant. A quantity (not normally subject to change) held in a computer so that it is available as data for the object program.

Contents. The data held in an area of a store.

Control.
(noun). The part of a computer containing the *instruction register* which controls the sequence of which program instructions are acted upon.
(verb). To direct the sequence of carrying out programmed instructions in a computer.

Control Bus. The electrical route taken by signals from the control logic in a computer, e.g. to fetch the next program instruction from memory.

Control Field. The location in a larger string of data of one or more bits or character whose presence or absence affects the significance or treatment of the rest of that piece of data. A typewriter keyboard offers a familiar analogy, where depression of the *shift* key alters the effect of pressing alphabetic keys.

Control Line. Synonymous with *control bus.*

Control Panel. The part of a computer where switches are grouped, allowing the user to monitor and modify the state of processing. Some visual display (e.g. of the bit pattern in a register) is normally provided, either by a row of lights or calculator-type display or it may use the computer's regular video screen.

Control Total. A means of verifying that a batch of numeric data has been entered correctly. The total of that data is entered and compared, by program, with the total accumulated in the computer as each item was entered. If the totals do not agree an error message is displayed so that the data can be rechecked.

Control X. The simultaneous depression of these two keys *(control* and *X)* on a full ASCII Keyboard normally generates the command for 'delete the current line'. *Control* with certain other alpha keys has special significance, for instance *Control G* will sound bell or bleeper, if either is fitted to the terminal.

Conversational Mode. A method of operation in which the user obtains a direct response to each input message, sometimes including a prompt for the next message expect by the program.

CORAL. Computer On-line Real-time Applications Language.

Correspondence Code. A bit code, similar to ASCII but not compatible, adopted by IBM for early terminals.

Core. (Ferrite core memory). An early form of random access memory which is non-volatile but bulky and expensive to manufacture.

Corrupt. To spoil a record of data or program. Corruption is distinguished from erasure because data is left which may look correct but is in fact worthless and could be positively harmful.

COS. Cassette Operating System – a computer operating system loaded from a magnetic tape cassette.

CPS. Characters Per Second, used especially of the rate at which output may be displayed or printed.

CPU. Central Processing Unit.

CR. Carriage Return.

CR/LF. Carriage Return and Line Feed: the two functions are sometimes combined to reduce the number of key depressions required of the operator.

Cross-talk. Spurious or unwanted signals picked up by interference from another circuit: similar to a crossed line in telephone conversations.

Cross-assembler. A program *assembler* used on a computer other than the one on which the object program is to be run.

CRT. Cathode Ray Tube (abbreviation). Often used to signify a *cathode ray tube* display terminal.

Crystal. A piezo-electric material cut to resonate at a stable predetermined frequency under the stimulus of an electronic oscillator. This principle is used to control the *clock frequency* of a computer.

C/S. Cycles per Second (abbreviation). Synonymous with the newer term *Hertz.*

Cursor. The screen character (typically an underlining dash) in a video display which indicates where the next character is due to appear if a suitable key is pressed.

CUTS. Computer Users' Tape Standard. A standard for transferring data to a computer from a magnetic tape cassette. Where different computers conform to the same standards a cassette written by one can be read by another, provided other details are also compatible, such as programming language if it is a program on the cassette.

Cybernetics. The study of both control and communication in machines, men and animals.

Cycle.
(1) The interval in which a set of events is completed. Normally used in the context of a set of operations repeated regularly in the same sequence, though possibly with minor variations.
(2) In tape storage: the period in which a tape can repeat its useful function. For example, a tape may enter a grandfather-father-son system as a "son". The following day (in daily processing), with the creation of a new "son" it will become a "father"; and the day after, a "grandfather". After discharging its "grandfather" function it will be available for erasure and re-use, perhaps to enter the next cycle as a "son".

PERSONAL EXPERIENCE

G. D. Hayes

Computers have featured in my life for several years now, since I found an NCR 4100 fascinating enough to cause me to fail my degree, due to my constant presence in the computer room, and absence from lectures. Since then, my computers (courtesy of the various companies for which I have worked) have nearly all been IBM 370 and 360 machines, of many different sizes, configurations, and operating systems.

Since the 4100 days, I have wanted my own machine. However, the one-woman Hardware Expenditure Vetting Committee (my wife) is unaccountably not in favour of the multiple mortgages and structural alterations required to install even a small 360 in the lounge. I have toyed with ideas of a commercial mini, but my experience of the leading varieties is one of lack of robustness, and susceptibility to power fluctuations and static. User serviceability does not seem to be one of their strong points either, and maintenance contracts tend to be a rather heavy monthsly outlay.

Two years ago I discovered the existence of the Amateur Computer Club. Off went my cheque, and back came ACC newsletters, with a design for a TTL mini, the WB–1. Not big, at 256 bytes, to a man used to upwards of half a megabyte, but cheaper than a 360, at about £50. There was the added bonus of knowing what the beast was doing when it was running, from a hardware standpoint, thereby improving maintainability when it (inevitably) went down.

Even so, was 256 bytes a usable size? The obvious way to find out was to write a program or two, and see if they would go in. This turned out to be a problem, as no program came immediately to mind. At the office, one always works to some kind of specification, even if it is only the fact that one has recognized a need to prevent the operator from doing something silly like starting the system on the wrong date. The need is there, and the program evolves from that. With a hobby machine, and that only a vague idea in the back of the mind, no need was immediately obvious.

Back to college, then. What did we do there? Prime numbers, factorials, Newton-Raphson, real and imaginary roots, circuit design, and pi to a lot of places, plus sundry other useful programs. Now, the pi program used regularly to run out of storage, and the 4100 had more than 256 bytes. The circuit design routines used a plotter, and under the circumstances that was an attempt at galloping before crawling had been attained. Newton-Raphson and the roots evaluation needed floating point and hardware divide/multiply to be viable. So we were down to prime numbers and factorials, and evaluation showed that these would go into 256 bytes with room to spare. On this dubious basis, the decision was made to go ahead, and many parts were ordered.

A few years ago, before I allowed my licence to lapse, I was a radio amateur. All my equipment was home-brewed, and full of big valves. 6V6, 803, 5U4 – it was almost possible to solder these (or rather their sockets) with a red hot poker. When the 7400s started arriving, they were impossibly small, with far too many pins. As well as this, the memory chips were labelled 'Static-sensitive – handle with care'. If there is one thing guaranteed to cause a normally healthy hand to grow twelve thumbs, it is any device labelled 'handle with care'.

But slowly, a 3-board mares-nest began to take some sort of shape, and as each board was tested and minor bugs eliminated, it became clear that no major mishaps had occurred. Only one defective IC was found, and the suppliers were very helpful and exchanged it although it had been soldered in. One apparently serious problem showed up when testing the arithmetic board. It would not get its sums right, although logical operations worked fine. Bit designation was the answer. In IBM terms the most significant bit of a byte is, (illogically), bit zero. On the WB–1 and commercial minis, bit zero is the least significant bit, two to power zero. Turning the display upside down worked wonders.

Finally it was all hooked together, and, with only a minor protest from the slightly cooking power supply, fired up for the first time. The first program was entered: L1 GTO L1. And as the comment in the ACC newsletter said, for ever! The run light came on and stayed on. It looked as though we were in business, and a testing session with programs somewhat more complex than the first one showed that the beast would actually compute.

But what a different world! With a 370, you press the blue LOAD button, REQUEST, and after a few typed responses you have a system which will run not only your job, but several others concurrently, and talk happily to disks, tapes, cards, printers and other peripherals with minimal impact on the programmer. With WB–1, it took slightly longer to enter one byte via the console switches than 370 takes to load a 60k supervisor. At this time, the home computing facility consisted only of the CPU, with eight LEDs for data output, and eight toggle switches for input. Peripherals were in the realms of fantasy.

Nevertheless, the luxury of coming home after a hard day's programming, to a hard evening's programming, was superb. The original factorial and prime number programs were laboriously loaded in, and after extensive debugging, made to work, reading out their answers on the console lights. Extended multiply and divide routines made their appearance, at great cost to the precious bytes. Programs became more adventurous and complex, although not particularly useful. Size was clearly a restriction.

To alleviate the problem of shoe-horning programs into the 256 byte memory, it was decided to extend the instruction set. The original WB used a single accumulator, and to move from X to Y two 2-byte instructions were necessary. A minor change to the control circuitry allowed a new set of three-byte instructions to carry out storage to storage operations, and this was a great help.

The first few months of operation taught me a lot. I had gained a great deal more by building the machine myself than if I had bought a commercial unit. I actually understood what was happening in the circuitry when it was running, which gave a pleasant feeling of satisfaction. With a manufactured machine I would not have been in a position to do very much about an instruction set with which I was dissatisfied. The flexibility from that point of view was a real bonus.

It was also interesting to see just how bad my programming was with the unfamiliar assembly language. Bugs which I would have spotted in seconds in an IBM assembly took all evening to correct, although one problem turned out to be due to a condition bit being set to the inverse of its original specification. As understanding increased with use, however, the assembly techniques became more sophisticated, and some of the solutions forced on me by lack of storage have also, due to their elegance, found a place in my day-to-day commercial work. The only drawback is a tendency for WB–1 assembler instructions to be input to the IBM assembler, which does not like them at all! I do, however, have a WB–1 assembler which runs on a 370, but as it takes 44K of storage there is not much chance of transferring it to the home machine for a while. At present, most assemblies are hand-decoded.

The little machine's first peripheral was a keyboard/display. This was (and is) a simple octal device, but was an immediate success. Interfacing proved reasonably easy, although the way in which WB–1 accesses storage made a read/write interface somewhat more complex than a read-only or write-only port would have been. Anyway, it worked, and entering programs was considerably eased. A small, eight-byte bootstrap was placed in storage via the switches, and the keyboard could then be used to enter the required program. This was much faster than the original method, and meant that small demonstration programs could be placed in the machine before a visitor had a chance to become bored. The display was helpful as well, since results could be output and held independently of machine operation. Previously, display could only be effected by stopping the machine, and examining the console lights.

At this time the relative ease of data entry led to the first domestically useful program, the timer. The required time period was entered on the keyboard, and a count-down ensued, culminating in the sounding of an alarm when the interval had ended. Computer-timed eggs, however, taste much the same as those cooked under normal conditions.

After the keyboard, mass storage. A totally non-standard cassette interface system was designed and built, and, to some amazement, actually worked. It could scarcely be called fast, but it was quicker and easier to use than the keyboard. It led the way to a rudimentary operating system, consisting of a group of services (keyboard input, cassette read, cassette write, and storage display), entered by a common entry point, and selected by keyboard input. At present, software is used for serial/parallel conversion, so loading the system means entering the keyboard bootstrap via the switches, keying in a simplified cassette bootstrap, and only then reading in the main system. Once that is in, however, the cassette may be used to load programs and store output, and the machine begins to feel like a real computer. It even boasts a relocating loader.

So what of the future? The immediate prospects are of little activity for about six months, as the summer always means a constant battle with the pigeons, caterpillars, and weeds who believe that they have more right to my vegetables than I do. At the moment, the computing complex has a teleprinter, but the interface is causing problems, mainly due to transients derived from the selector magnets. I don't suppose that that will be sorted out until the autumn. Similarly, all the chips have been bought to allow a storage upgrade to 1K, but that too must wait. However, in its present state, the machine plays a passable game of noughts and crosses, as well as mastermind, and will even tell you the day of the week on which you were born. Still not particularly useful, admittedly, but quite satisfying in its own way.

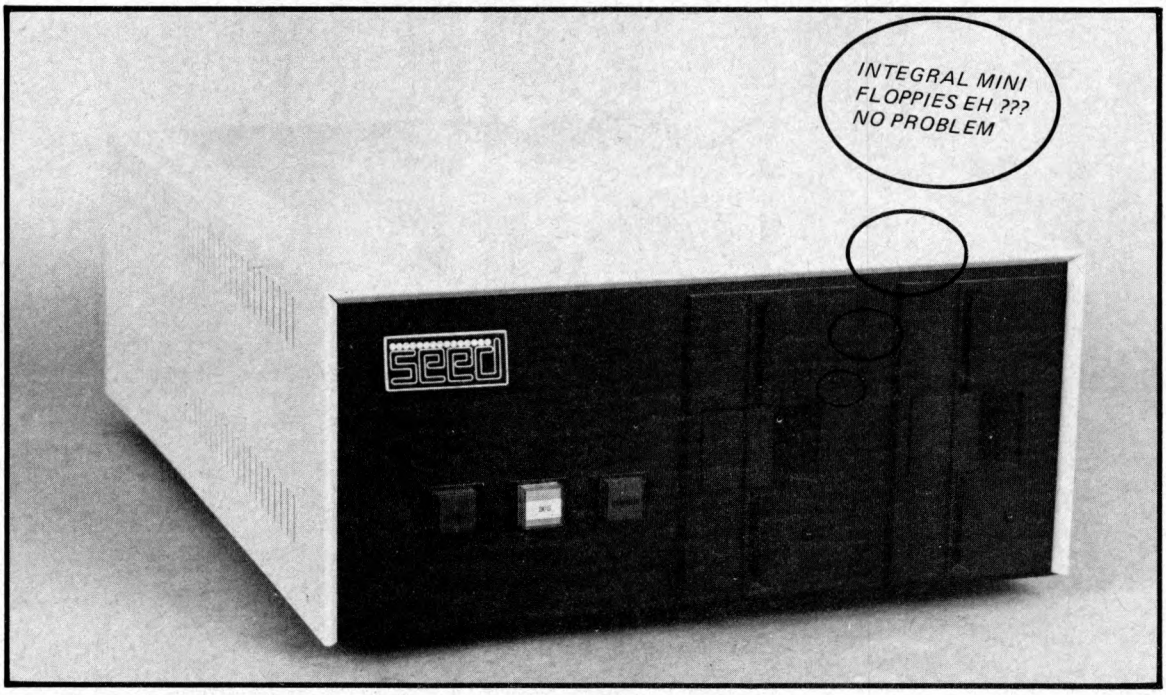
If I were starting from scratch, and designing my own machine, I think I would still use TTL. The flexibility it allows, with regard to enhancements, must surpass the micro-processor type of machine. The speed of a micro is probably higher, but it seems to me that it is debatable whether a domestic machine needs to head into the MIPS fraternity. However, I now realize that there are a number of facilities which a truly useful home computer should have. More storage, obviously, and an instruction set which allows operations between storage locations. Although not a problem to date, an interrupt system to allow a program running in the machine, such as a TV game, to be temporarily shelved while another, asynchronous function is serviced, as for instance the famous central heating control. Another facility, closely associated with an interrupt system, would be a programmable real-time clock, useable for the simultaneous and accurate timing of a number of events.

If the machine is to be used for arithmetic operations to any degree, hardware multiply/divide is mandatory. The software simulation of these is expensive, both in terms of execution time and, more particularly, storage use. Financial planning and accounting, car statistics and maybe the childrens' homework are typical areas in which a home computer can earn its keep, and all will benefit from the implementation of the multiply and divide instructions.

Possibly, in a year's time, the WB–1 will have most, if not all, of the above features. Certainly, it should be a fairly powerful machine, and, I hope, will be beginning to take its rightful place in the household. I envisage applications such as those outlined above, as well as classic roles like generalized documentation of correspondence, an intelligent calender/diary, and centralized electrical monitoring and control. The machine already plays some games, and interfacing with the Teletext decoder will allow a colossal flexibility in display of the output from these programs. A frequency-selective filter matrix, fed from the audio system and driving analogue to digital converters feeding the computer, will afford programmable, dynamic video displays in true sympathy with music being played. All this and central heating control! Who needs a 360?

seed

STRUMECH ENGINEERING ELECTRONICS DIVISION



THE **SYSTEM ONE** COMBINES THE BEST OF

MSI 6800 PROCESSOR WITH 16K RAM.
(EXPANDABLE UP TO 56K)

SSB MINI FLOPPIES *(SINGLE OR DUAL. 80K PER DISK)*

THE SYSTEM COMES COMPLETE WITH BASIC INTERPRETER AND DISK FILE HANDLING SOFTWARE.

seed

STRUMECH ENGINEERING ELECTRONICS DIVISION
Portland House Coppice Side Brownhills 4321

KEEPING AN ORDERLY HOUSE

Monitors for Microprocessors

A. Clements, Department of Computer Science, Teesside Polytechnic, Middlesbrough.

PCW *We urge even beginners to read this article, it contains clearly explained ideas. And intending authors: this article can be studied to your profit!* PCW.

Introduction

Microprocessors are used in a wide spectrum of applications from intelligent test equipment, to the poor man's digital computer, which offers many of the facilities of a mainframe computer, at a fraction of the cost.

All microprocessor systems, without exception, are composed of two fundamental components – the hardware and the software. The hardware component of a microprocessor system consists of all the physical components which make up the system, while the software component is the program which resides in the microprocessor's memory. Although hardware and software are different entities, they are often interchangeable – for example, multiplication may be performed by means of a special electronic circuit (hardware), or by means of a program (software). The hardware part of a microprocessor system consists of the CPU itself, control circuits (e.g. clock, DMA, interrupt systems), memory components, and interface circuits.

In recent years not only has the range of hardware components increased, but often the complexity of an interface or control integrated circuit rivals that of the microprocessor itself. For example, Motorola has recently introduced a CRT controller which performs many of the functions of a VDU so that a microprocessor may readily be interfaced to a rasterscan display (TV). Furthermore, this device has 19 on-chip registers which allow the user to specify a wide range of display formats (number of characters per line, number of lines per page etc), and even has facilities for a light pen. The most important feature of hardware is that components can be bought off-the-shelf at prices which bear little or no relationship to their complexity, are usually very reliable and come with detailed specification sheets and application notes.

The software components of a microprocessor system cannot usually be bought off-the-shelf for the following reasons:

(i) Many different microprocessor systems use the same basic hardware components (CPU, buffers, gates, memory components etc), but each new use of a microprocessor requires new software to be written. For example, consider the range of software required by the following three systems: a point-of-sale system in a supermarket; a fuel control system in an automobile designed to optimise the air/fuel mixture to reduce the emission of carbon monoxide; and a small general purpose computer designed to run programs written in BASIC. *Although the above systems may use very similar hardware, the software must be tailormade for each system.* Clearly, software cannot yet be mass produced.

(ii) Software is highly system dependent at the machine code level. Although the same UART can be used in either a 6800 or an 8080 system, the object code used by a 6800 is different to the object code used by an 8080. This makes the use of standard software difficult, unless the source program is written in a high level language and a compiler used to generate object code for the target microprocessor.

(iii) The state-of-the-art. The theory, design and testing of hardware systems has reached a more advanced state than the theory, design and testing of software. For example, the TMS 9900 is one of the most sophisticated microprocessors, with a sixteen bit data word, on-chip multiplication and division, and a very advanced architecture, yet it costs only £50, while a compiler for PASCAL can cost £3,500.

For the above reasons, the hardware design of a microprocessor system may be relatively straightforward, but the design of the software a costly headache. The hardware designer has, at his fingertips, a whole range of very

sophisticated building blocks from which he can put his system together. The programmer is not so lucky, he must construct his program line by line. He cannot, yet, obtain software modules with which to construct his program.

Approaches

There are several approaches to the design and testing of software. One possible approach is to use a time-sharing computer terminal and rent time on some large, central mainframe computer. In this way a program can be entered, edited, assembled, and then tested by simulating the system being designed, so that the designer can find out how the program would behave if it were resident in his system.

An alternative approach is to develop the software in a system which *uses* the microprocessor to be employed in the final system. The designer is, in effect, building (or buying) a general purpose computer on which he can develop and test his software. This computer consists of the hardware configuration required by the application for which the microprocessor is intended, plus an interface to a VDU or teletype which allows programs to be entered, tested, modified and stored. *The simplest program which will enable the designer to perform the above functions is called a* **monitor.**

A monitor may be thought of as a very simple operating system. If the monitor is a program, resident in the development system, and required to allow the designer to create his own special purpose programs, where does the monitor come from? Major microprocessor manufacturers, have been quick to realise that small manufacturers, educational institutions, and hobbyists do not have access to the facilities needed to design their own monitors, and have provided off-the-shelf monitors for use in development systems.

The Architecture of a Monitor

A monitor is a program, resident in read only memory, whose minimal function is to enter a program into the microprocessor's random access memory and then to transfer control to that program, that is, to enter and run a program. Although a monitor can be designed to allow the program to be entered by means of binary switches on a front panel, the availability of teletypes, VDU's and cassette recorder interfaces means that very few monitors are designed to be used in conjunction with a front panel.

Most monitors communicate with the teletype or VDU by means of an asynchronous, serial data link. That is, data is transferred between the input/output device and microprocessor as a *series* of electrical pulses. These pulses are not synchronised at both transmitter and receiver to some master clock. *Instead,* each group of pulses (typically 10 or 11) begins with a start pulse which serves as a *reference* to the following pulses. This synchronism need only be maintained over a duration of up to 11 pulses. The pulses are normally transmitted in one of two forms: a 20 milli amp current loop or bipolar levels of +12v or −12v. The current loop system is normally used with teletypes which operate on a purely electro-mechanical principle, and the ±12v system is used mainly with VDU's, and is called the V24 or RS 232C interface. The serial data is transmitted in the ASCII (American Standard Code for Information Interchange) format, which uses seven bits to encode the 2^7 = 128 characters used by VDU's. These characters include the set of upper and lower case letters, numbers, special signs and control characters. Most teletypes, however, use a six bit subset of the ASCII code, as they do not display the lower case letters. Along with the data a parity bit is usually transmitted; that is, an additional bit which makes the total number of ones in the transmitted data word even. This acts as a partial safeguard against transmission errors due to noise picked up by the data link.

In general, the serial data word transmitted between microprocessor and terminal contains a start bit, seven or eight bits of data, an optional parity bit which may be odd or even, and one or two stop bits. This places a severe restriction on the input/output *device* because it must transmit/receive data in the format required by the *monitor.* This restriction means that if the data link is designed to work with 7 data bits, as it usually is, nine hole paper tape cannot be read and data cannot be stored on cassette in the efficient form of one character per byte.

Motorola's MIKBUG monitor stores data on paper tape (or cassette) in the form of two hexidecimal characters per byte. This system is clearly very inefficient, especially when it is realised that the hexadecimal code, 0, 1, 2, . . . 9,A,B,C,D,E,F, has a 50% redundancy and yet gives no additional error protection. Some of the more sophisticated monitors allow the user to specify the serial data format from the keyboard. Of course, the user must still have the interface required by the monitor before he can define a new interface.

All monitors place important restrictions on the systems design. Each monitor resides in a ROM which must occupy some of the available memory space. Furthermore, each monitor needs some random access memory to hold temporary variables. The location of the monitor itself, i.e. the address of the monitor, is determined by the way the microprocessor starts after power is applied or the reset button is pushed. For example, when the Motorola 6800 is reset, the (16 bit) program counter is loaded with the (8 bit) *contents* of memory locations $FFFE_{16}$ and $FFFF_{16}$. For this reason any monitor in a 6800 system is located so that it will respond to addresses $FFFE_{16}$ and $FFFF_{16}$. The location of the scratchpad RAM used by the monitor can be anywhere within the available memory space. The actual location of the RAM is determined only by the person who wrote the monitor. For example MIKBUG requires 128 bytes of RAM at $A000_{16}$. This may be a severe limitation if the user requires a large amount of contiguous RAM for his own program.

Range of functions

The range of functions provided by monitors is varied, but two functions are common to all monitors. The monitor must be able to input data from a peripheral, store it in a specified area of RAM, and then transfer control to the program which has been loaded. The data which is read by the monitor is usually stored in the following form on paper tape or cassette:

CHECKSUM DATA ADDRESS NUMBER OF BYTES HEADER

The **header** is a character, or group of characters, which signifies the *start* of the data block. The monitor ignores all data until the header is found, a facility which allows the user to precede the data block by the name of the program without the monitor trying to interpret the name as part of the data block.

Immediately following the header is the **byte count** which tells the monitor the length of the data block to be read i.e. address + data + checksum.

The **address** consists of two bytes which specify where, in memory, the data is to be placed. Two bytes are required to specify the address, because the address field is 16 bits wide, allowing 2^{16} = 65,536 possible memory locations to be specified.

The *data following* the address consists of the actual data to be loaded into memory, starting at the specified address and then loading successive bytes of data into consecutive memory locations.

The last byte in the data block is the *checksum,* which is the one's complement of the summation of the 8 bit bytes following the header. That is, when the tape

"Bo!%@* the buffer speed my sales ledger was up in 20 minutes."

What is important is whether the system works for you and how long it takes to get you on the air.

We are distributors of the full range of apple products and other peripherals with software back-up.

Collins Consultants

For information or demonstration ring Collins Consultants, Tel: (02816) 2572 or Templeman Software, Tel: (0789) 66237

EQUINOX 300

A powerful multi-user
multi-tasking
multi-language

16-bit microcomputer time-sharing system

supporting
* BASIC
* LISP
* PASCAL
* Floppy Discs
* Hard discs

including a powerful Text Formatter, Assembly Language Development System and disc based Sort utilities.

Priced from under £5,000

Write or phone for further information

EQUINOX COMPUTER SYSTEMS LTD.
32-35 Featherstone Street,
London, EC1Y 8QX
(01-253 3781/9837)

is first generated, the checksum is set to zero and each byte, after the header, is added to the checksum. Only the least significant eight bits of the checksum are used in addition, so that the checksum remains eight bits long no matter how many bytes are added to it. Before the checksum itself is written on the tape, the checksum is complemented so that the O's become I's and vice versa.

When the tape is read by the monitor, the sum (again neglecting overflow) of all the bytes in the data block, after the header, is formed. The value of this sum should be FF_{16} = IIIIIIII. If this value is not obtained the tape has been incorrectly read. The checksum is therefore a device to allow the detection of errors when reading the tape. Such a facility is vital when data is stored on a cassette.

After a program has been loaded, it may then be executed by jumping to its entry point, that is, the program counter must be loaded with the address of the first instruction in the program to be executed. This is often done by loading the index register with the entry point and then performing an indexed jump. Motorola's MIKBUG transfers control to the user's program by storing the entry point of the user's program on the stack and then executing a return-from-interrupt instruction. This has the effect of loading all the M6800 registers from the stack (except, of course, the stack pointer itself).

Some other functions provided by monitors

Dump/Punch This is the complement of load. After a program has been created, or an old program modified, it may be stored on paper tape or cassette in the form required by the load function. Almost all monitors have this function.

Examine memory location This allows the contents of a selected memory location to be examined and, if necessary, modified. This function is used by all monitors, although its sophistication varies widely from monitor to monitor.

Move a block of memory This allows a given block of data to be moved from one region of memory to another. (More precisely, the data is copied from one region to another). This function is sometimes referred to incorrectly as relocation. *True* relocation not only involves moving the data but also recalculating the address of jump and branch instructions. The move function is not provided by all monitors, and is, in the author's opinion, of minimal use.

Preset a block of memory locations This instruction allows a block of contiguous memory locations to be set to a given value. Typically, the block of locations may be set to zero to clear the memory. Again, this function is of dubious value.

Examine registers This function allows the contents of the microprocessor's registers to be displayed on a teletype or VDU. The nature of this function varies from microprocessor to microprocessor. This function is usually used to debug programs.

Hexadecimal arithmetic Some monitors allow the direct evaluation of simple arithmetic expressions using hexadecimal arithmetic, at the keyboard. This facility allows the calculation of offsets, which is useful in determining relative addresses. Otherwise, this is an unimportant function.

Locate a byte/word within a given range Few monitors provide this function which allows the user to search through a specified area of memory for a *given* byte or word (one word = two bytes). This facility is often useful when modifying programs because a particular op-

code or address may be speedily located without having to step through memory locations.

Set a breakpoint A breakpoint is a means of halting the execution of a program at a specified address. When the breakpoint is encountered, that is, the *program counter* contains the address of the breakpoint, the contents of the microprocessor's registers are displayed on the VDU or teletype. This is a very important facility as it allows the programmer to examine the contents of the microprocessor's registers *during* the running of a program. Some monitors allow several breakpoints to be set. The ability to set breakpoints is found only on the more sophisticated monitors.

Trace A few monitors allow the execution of a program in the trace mode when, after each instruction in a program has been executed, the contents of all the working registers are displayed. The trace mode is, therefore, similar to the use of a breakpoint, except that the breakpoint is automatically moved through the program *during its execution.* The trace facility is usually implemented by hardware techniques, which cause each instruction to be interrupted.

A list of the functions performed by a monitor is not always a good guide to the suitability of the monitor in a given application. To understand this point it is necessary to examine a particular monitor. One of the oldest and most widely used monitors is Motorola's MIKBUG which, although designed for their MK 6800 DI introductory kit, has become a standard for 6800 systems used by many educational establishments and hobbyists. This has led to the production of much 6800 software designed to operate in a 6800 environment. That is, the software uses subroutines within the MIKBUG ROM, and uses the same scratchpad RAM as MIKBUG. Some 6800 software uses the MIKBUG input/output interface, a peripheral interface adapter located at 8004_{16}.

One of the advantages of MIKBUG is that its subroutines can be called from the user's *own* programs. The most two important subroutines in MIKBUG are the **character input** and **output** subroutines.

The input subroutine fetches a character from the keyboard and puts it in the A register, and the output subroutine causes the character whose code is represented by the contents of the A register to be displayed on the system console.

Other useful subroutines allow data to be entered (or printed) in hexadecimal format, and strings of characters to be displayed on the console. These facilities are very important to the beginner because they allow him to write programs without having to worry about the design of his own input/output routines.

The great disadvantage of MIKBUG is its inflexibility. The input/output routines used by MIKBUG convert the parallel data in the A register to a serial format by shifting bits out of a PIA, using a hardware timer to determine the interval between successive bits and hence the bit rate of the serial data. Three disadvantages follow from this:

1) The input/output interface is unusual and must be built into any system using software which employs MIKBUG input/output routines.
2) The CPU cannot be used for the purpose of computation until *after* a character has been printed. When an asynchronous interface device is used to output characters, the CPU takes no part in the output process once the character to be printed has been written into the interface device.
3) While the CPU is outputting a character, bit by bit, the subroutine does not check to see if any new data has arrived from the keyboard. This means that input and output cannot take place simultaneously, and that output cannot be interrupted, which prevents a break in being made.

A further disadvantage of MIKBUG is that when erros are detected in a MIKBUG subroutine, control is passed back to MIKBUG. If a user has written a program to read data in hexadecimal format, from a cassette system, using the MIKBUG subroutine BYTE, located at $E055_{16}$, no problems will occur as long as valid hexadecimal characters are read off the tape.

As soon as a non-hexadecimal character is detected by BYTE, control is passed to MIKBUG. This is bad enough, but MIKBUG now regards further characters coming from the tape as valid MIKBUG commands. A better way to design an input routine is to set an error flag (usually the carry bit) when an input error is detected. In this way a subroutine cannot pass control to the monitor, if an invalid character is encountered.

FBUG

FBUG is the author's own monitor which is designed to operate in a 6800 system. The monitor occupies 1024 bytes of ROM from $FC00_{16}$ to $FFFF_{16}$, with a 6850 ACIA located at $F800_{16}$, and 128 bytes of scratchpad RAM at $A080_{16} - AOFF_{16}$.

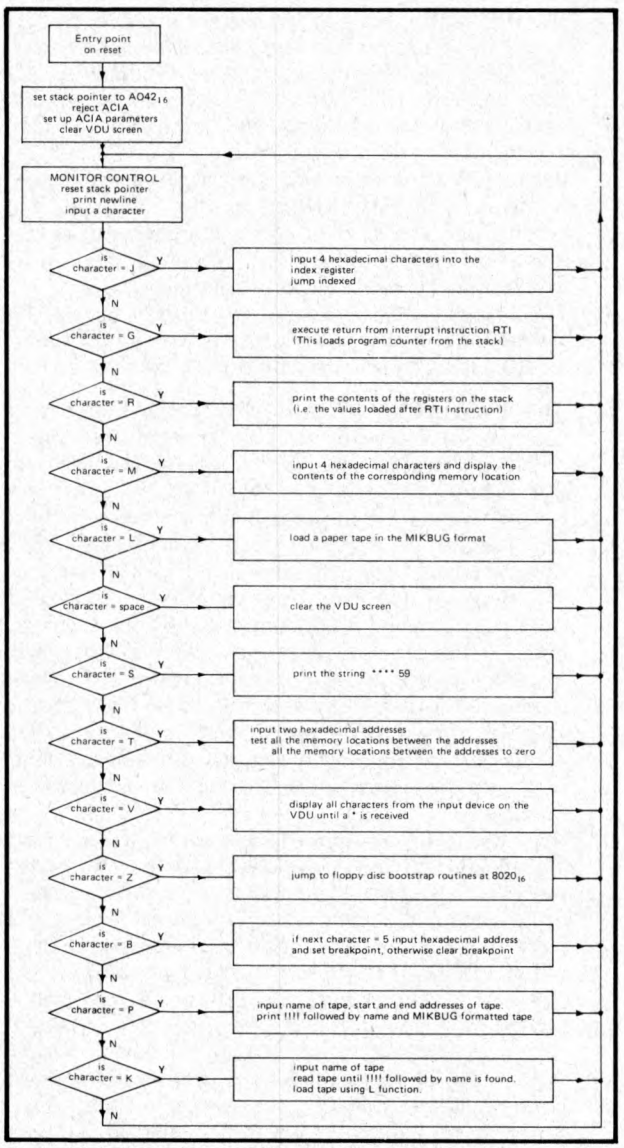


Figure 1: Functions performed by the monitor

The basic functions provided by FBUG are described in Figure 1, each function being activated by typing a single character on the console. The most significant features of FBUG are as follows:

1) Two methods of **entering** a user program are provided. Typing a J followed by a four character hexa-

decimal address causes a jump to the address, and typing a G causes a return from interrupt instruction to be executed which loads the contents of $A048_{16}$ and $A049_{16}$ into the program counter. The G command allows compatibility with MIKBUG oriented software.

2) The **memory examine** function, M, is one of the most frequently used monitor commands and has been made very versatile. After typing M followed by the four character hexadecimal address of the location to be examined, the monitor displays the contents of the location as two hexadecimal characters. If any character other than a B, I, N, space, or carriage return is typed, the monitor prints the address of the next location and its contents on a new line. If a B is typed the monitor prints the address and contents of the previous location, which allows the correction of errors without leaving and re-entering the M command. If an N is typed, a new four character address may be entered which becomes the current location and its contents are printed. This allows non-consecutive memory locations to be examined. If a carriage return is entered, control returns to the monitor. If a space is entered, the contents of the current location may be modified by typing two new hexadecimal characters. The address and contents of the next location are then printed. If a slash, /, is typed the monitor assumes that the data displayed in the current memory location is a valid operation code, and prints, on the same line, the operand, which may be zero, one, or two bytes, depending on the length of the instruction. This facility allows a program to be displayed in the form of one instruction per line, which makes it relatively easy to interpret.

3) The **L command** is used to load a paper tape in the format described earlier in this article, and is functionally equivalent to the MIKBUG L command.

4) The **K command**, which is used to load data from a cassette, prompts the user with a question mark, and expects the name of the required program on the tape to be typed in followed by a carriage return. The tape recorder is then switched on, in the play-back mode, and the monitor will ignore all data until a tape header, !!!!, followed by the name of the required program is encountered. Control is then passed to the subroutines used by the L commnad and the program read as if it were a paper tape. This facility was added because it proved difficult to position the tape precisely at the start of the required program. Failure to do this often resulted in the end of some other program on the tape being read and overwriting some important program in memory.

5) The **input/output subroutines** used by FBUG have been made as flexible as possible. While writing the monitor it became evident that at some later date a different input/output system might be required to work with the existing monitor. How was this flexibility achieved? Before the monitor attempts to input or output a byte, it examines a byte called MODE which is stored in the scratchpad RAM. If bit O of MODE is unset (i.e. b_0 = O) the normal output routine is used. If, however, bit O is set, a jump to location $AOCO_{16}$ is executed. Thus, to use a new output device, with the existing subroutines of FBUG, a jump to the subroutine which handles the new output device is placed at location $AOCO_{16}$ and bit O of MODE is set to 1. A similar procedure applies to the input subroutine except that bit 1 of MODE is tested, and if set a jump to $AOC3_{16}$ is executed. The input/output subroutines of FBUG are illustrated in figures 2 and 3 by means of a flow chart and assembly language listing respectively.

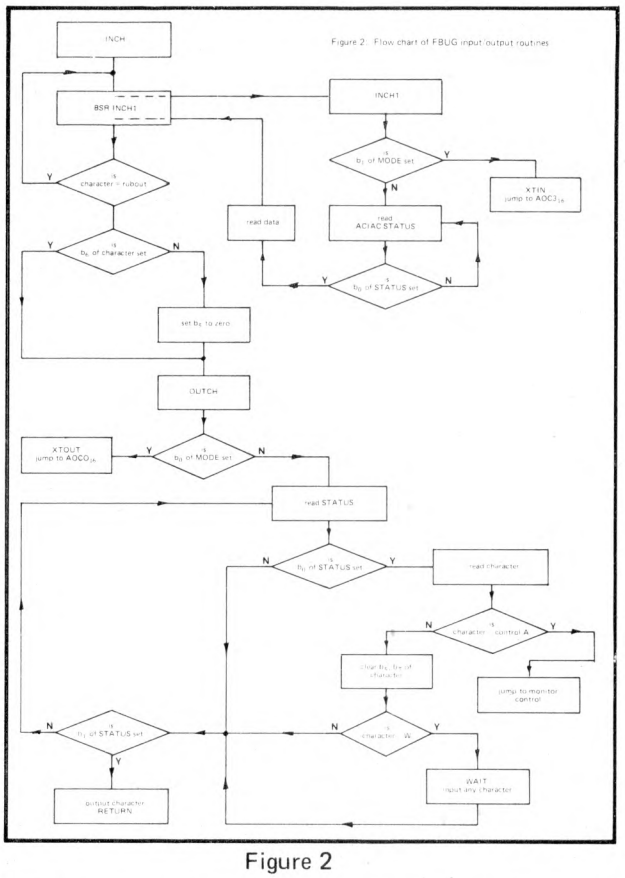


Figure 2

```
INCH1    LDA A MODE          TEST FOR EXTERNAL INPUT
         AND A #$02
         BEQ *+5
         JMP XTIN
INCH2    LDA A ACIAC         LOAD STATUS OF ACIA
         ASR A
         BCC INCH2           RECEIVE NOT READY
         LDA A ACIAD         INPUT CHARACTER
         RTS
INCH     BSR INCH1           GET CHARACTER
         AND A #$7F          REMOVE BIT 7
         CMP A #$7F          WAS IT RUBOUT?
         BEQ INCH            IGNORE RUBOUT
         BIT A #%01000000    TEST FOR BIT 6=1
         BEQ OUTCH           BIT 6=1, ECHO CHAR AND RETURN
         AND A #%01011111    BIT 6=0, CLEAR BIT 5
OUTCH    PSH B               SAVE B REGISTER
         LDA B MODE          TEST FOR EXTERNAL OUTPUT
         AND B #$01
         BEQ *+6
         PUL B
         JMP XTOUT
OUTC1    LDA B ACIAC         LOAD STATUS OF ACIA
         ASR B
         BCC OUTC2           TEST FOR INPUT
         PSH A               SAVE A REGISTER
         LDA A ACIAD         PICK UP INPUT
         CMP A #1            BREAK IN?
         BEQ START           YES : JUMP TO STAND
         AND A #%01011111    REMOVE BITS 5,7
         CMP A #'W           WAIT?
         BNE *+4
         BSR INCH1           INPUT ANY CHARACTER TO CONTINUE
         PUL A               RESTORE A REGISTER
OUTC2    ASR B               TEST FOR OUTPUT READY
         BCC OUTC1           TRANSMIT NOT READY
         STA A ACIAD         OUTPUT CHARACTER
         PUL B               RESTORE B REGISTER
         RTS
```

Figure 3

Data

Two subroutines may be used to input data. INCH1 simply inputs an eight bit byte without echoing it (i.e. it is not automatically printed on the console). INCH uses INCH1 to input a character and then removes bit 7 from the character, checks for the rubout character (7F), converts lower case characters to upper case characters, and prints the character on the console. It is necessary to convert lower case characters to upper case because most software is designed to operate with the upper case character set which is produced by the teletype, but VDU keyboards produce lower case characters unless the shift key is pressed. The advantage of having two input

routines is that pure binary can be read from tape without having to split a byte into two hexadecimal characters.

Two useful features of the **output routine** are the ability to break in, and the ability to temporarily suspend the printing of data. At the beginning of the output routine the ACIA is tested to see if anything has been received from the keyboard. If any character other than a control A or W has been typed, it is ignored. A control A causes a return to the monitor control, and a W halts output until a second character is typed which allows the operator to read the contents of a VDU screen before it is overprinted by new output.

Postscript

The above monitor has been in use for several months and no bugs have been discovered. However, if the monitor were to be rewritten some new facilities would be added and some existing facilities removed. A major modification to the subroutines of FBUG would be to make them re-entrant. Although the present subroutines may be interrupted, they may not be called by the interrupting program because any variables being used by a subroutine will be overwritten when the subroutine is called by the interrupting program. This problem arises because the variables used by a subroutine are stored in fixes, or static, locations. To overcome this problem all variables used by the subroutine must be stored on a stack or somewhere within the calling routine, so that the subroutine may be interrupted and used without the pre-interrupt variables being destroyed. The subroutine is then said to be re-entrant.

Bibliography

ENGINEERING NOTE 100 (MOTOROLA LTD) This describes the 512 byte monitor, MIKBUG, and includes the circuit diagram of an interface, suitable for use by MIKBUG. A complete listing of MIKBUG, with comments, is given which provides the beginner with many useful programming hints.

SWTBUG USERS GUIDE (SOUTHWEST TECHNICAL PRODUCTS CORP) SWTBUG is a plug-in replacement for MIKBUG. This document describes the additional features provided by SWTBUG, which is a 1024 byte monitor. A full listing of SWTBUG is given, together with several useful memory-test programs.

RT/68MX SYSTEMS MANUAL (MICROWARE SYSTEMS CORP.) Like SWTBUG, RT/58 is a 1024 byte replacement for MIKBUG. The most important feature of RT/68 is its ability to operate in a real time multiprogramming mode. That is, RT/68 allows up to 16 programs to be in memory simultaneously, with control being passed from program to program. The RT/68 manual gives a full description of the monitor, full details of the multiprogramming mode, and a full listing of the program with detailed comments. An example of the use of the RT/68 is also provided. This manual provides a superb introduction to multiprogramming for the beginner, but it is expensive – £50 for the ROM and manual.

MEK6800D2 EVALUATION KIT II MANUAL (MOTOROLA) This manual is part of Motorola's evaluation kit. It is a complete description of the hardware and software provided by the kit. The monitor used, JBUG, is fully documented with detailed flow charts of the functions provided by JBUG.

PIPBUG APPLICATION MEMO SS50 (SIGNETICS) This describes Signetics 1024 byte monitor PIPBUG, and should be read in conjunction with applications memo SP50 which describes the hardware aspects of their evaluation board. SS50 provides a listing and a description of the functions of PIPBUG.

SC/MP INTROKIT USERS MANUAL (NATIONAL SEMICONDUCTORS) This describes both the software and hardware of National Semiconductors Introkit. A full listing of the 512 byte monitor, KITBUG, is given.

QBUG (QUARNDON ELECTRONICS LTD) Quarndon Electronics provide several different documents which describe QBUG. This monitor occupies 1536 bytes of ROM (1½K) and comes in two versions, one for the 8080 and one for the 2650 microprocessor. QBUG provides a wide range of facilities including the ability to specify hexadecimal or octal input/output. Quarndon do not supply a listing of QBUG.

AMSAT 8080 STANDARD DEBUG MONITOR (BYTE) This is the description and listing of a monitor for use with an 8080 system.

The
NORTH STAR HORIZON
Computer

HORIZON – a complete high preformance microcomputer system with integrated floppy disk memory

HORIZON is attractive, professionally engineered deal for business, educational and personal applications.

To begin programming in EXTENDED BASIC, merely add a CRT, teletype or other hard-copy terminal HORIZON–1 includes a Z–80A processor, 16K RAM, minifloppy disk and 12-slot S–100 motherboard with serial terminal interface, all standard equipment.

and software, too!

HORIZON includes the North Star disk operating system and full EXTENDED BASIC from diskette ready at power-on.

This Basic, now in widespread use, has virtually everything desired in a Basic, including sequential and random disk files, formatted output, a powerful line editor, strings, machine language call, and more.

OPTIONAL software (under CP/M) include – C–Basic Compiler/Interpreter Basic, Microsoft Disk Extended Basic, MAC Macro Assembler, Microsoft COBOL–80 and FORTRAN–80, and more.

QUALITY AT THE RIGHT PRICE:

HORIZON Z–80A processor board, RAM FPB, and micro disk system can be bought separately for either Z80 or 8080 S–100 bus systems.

NEW LOW PRICES

*HORIZON–1–16K: £1215 (£1075 kit)
*HORIZON–1–24K: £1325 (£1175 kit)
*HORIZON–2–16K: £1490 (£1330 kit)
*HORIZON–2–24K: £1595 (£1425 kit)

All prices are exclusive of VAT and carriage. Subject to change.

Full details: **PCW,** Interam Computer Systems Ltd., 59 Moreton Street, Victoria, London SW1V 2NY, Tel: 01-834 0261/2733

SIRTON PRODUCTS

COMPUTER INTERACTIVE DISPLAY
£97.50*

- 16 Lines of 64 Characters
- Computer Selectable Reverse Video on any Character, word, Line or Lines.
- Computer Selectable Flash on any Character, Word, Line or Lines.
- High Speed Page Write.
- Cursor Mobile in Four Directions. Blinks at 2Hz.
- Screen Scroll. • Screen Roll • Reverse Video.
- ELE – CLE – Home – Clear Screen.
- Responds as a TTY to CR & LF.

INPUT: 8-bit parallel or RS232/20mA serial.
OUTPUT: UHF modulator for domestic TV aerial connection or 1 vold P.P. composite video.
POWER: 120 or 240 V AC or 5 V DC @ 400 m A.
CASE: Two-part, size 8" x 6" x 2".

*Add £1 P & P + 8% VAT.

. . . . . .

ALSO: Touch-type keyboards from £38.50
Manual keyboards from £47.00

Send for our catalogue on JADE S.100 Bus Boards; also keyboards, mainframe, transformers etc. etc.

SIRTON PRODUCTS, 13 Warwick Road, Coulsdon, Surrey.
Tel: 01-660 5617

STAYING IN SHAPE

Michael James

Introduction

How do you write a program? Where do you begin? What sorts of operations and decisions do you use?

The chances are that your answers to these questions are vague because it is possible to write programs without worrying about how you manage to do it (rather like riding a bicycle). Programming is usually taught by a brief introduction to some programming language (BASIC, FORTRAN or ASSEMBLER say) and then practical experience. What this usually means is that once the details of the language have been mastered the student is expected to "know" how to program, and all that is required are a few "tricks" which can be picked up along the way (again rather like riding a bicycle). Structured programming (SP) is not about any particular programming language – it is a programming technique. Although some languages are more suited to SP than others, a knowledge of the science of programming can be used in any language to produce better programs.

Structure and Blocks

The basic principle of SP can be stated simply –

any program, no matter how complex, can be written using only a small set of "building-blocks" each of which has only one entrance and one exit.

A block is a recognisable type of program that is used fairly often. The most obvious example is a loop – the same operation being carried out over and over again until some condition is satisfied. In flow diagram terms (see PCW vol. 1, no. 1), which is perhaps the best way to think of it, a block is a collection of boxes joined by arrows in some standard pattern. The idea will become clearer after a few practical examples.

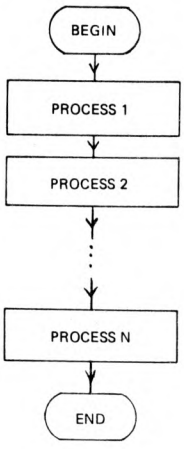


Fig. 1

The basic blocks of SP are

1) *The Sequence Block* which is the simplest of the blocks, being a collection of instructions obeyed one after the other.

It is important to realise that the processes may be any of the other blocks explained in this article.

2) *The DO WHILE Block* which repeats a sequence block as long as a condition, P, remains true.

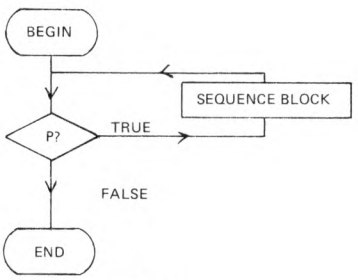


Fig. 2

Notice that the test is made before the sequence block is entered. Therefore it is possible to go through the DO WHILE block without ever executing the sequence block.

3) *The DO UNTIL Block* which repeats a sequence block until a condition becomes true.

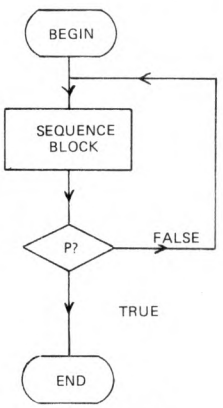


Fig. 3

Notice that, in this case, the test is made after the

sequence block has been performed – hence the sequence block has to be carried out at least once.

4) *The IF Block* which tests some conditions and carries out sequence block 1 if the condition is true and sequence block 2 if it is false.

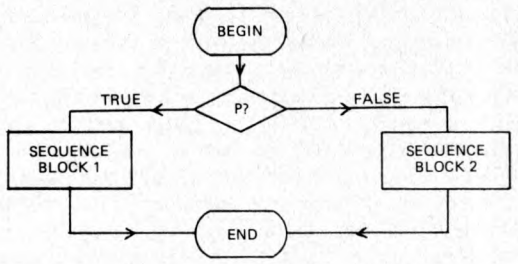


Fig. 4

Using just these four blocks - the Sequence, the DO WHILE, the DO UNTIL and the IF – any program can be written by joining END boxes to START boxes. It's rather like making a chain – each link being one of the blocks. In fact not all the blocks are strictly necessary, e.g. the DO blocks can be replaced by IF and Sequence blocks, but the given four have proved, in practice, to be the most useful. The subject of how few blocks or "structures" you actually need to be able to do anything is a fascinating subject in itself, but not really a practical concern and would lead us into the realms of Turing machines and the theory of computation.

A common feature of all the blocks is that, for each, there is only one START and one END box. This is a simple result of our one entrance one exit condition. Why is this one-way-in one-way-out property so important? Well – imagine that you are debugging a large program and that you are examining the contents of some variable at a particular point in the program. To be able to decide if it is correct you have to be able to predict what value it should have. How easy this is depends on how many ways there are to reach that point in the program. If the program is unstructured then there is likely to be more than one way and the problem is always which way. In *a structured program each piece of program can only be reached by one route!* Hence you have a much better idea of the values you expect to find at each point. This not only helps with debugging but makes a program less liable to have errors in the first place. Another advantage is that when the program is being changed (improved?), we only have to worry about the effect of the change in a small area of the program.

The most obvious feature of a structured program is that control does not pass up and down the program in the usual way. Therefore the unconditional GOTO or JUMP instruction is never used.* This fact was seized upon as being the essence of structured programming and for a long time people kept telling me that SP was "the art of not using the GOTO". Now, while it is true that the GOTO should never be used in an unstructured way, simply not using a GOTO does not make a program structured! This confusion caused many a programmer to despair at the difficulty of SP.

The other advantages claimed for SP are due to the use of standard blocks, making it easier for other programmers to understand and then update your programs. It is even supposed to be more enjoyable to write a structured program!

Pseudocode or a Real Language

Not all computer languages have to exist! As an intermediate step it is sometimes useful to write the ideas and solutions embodied in a flow diagram in a computer language that is a figment of one's imagination. The flow diagram makes no reference to any particular computer or computer language. The next step in the process is to code the program in some suitable language . . . the question is which? If we choose a traditional language such as FORTRAN or BASIC, both of which are popular and available, then we have to admit that our nicely structured flow diagram will be mutilated in translation by the inevitable tricks which we all employ in our favourite language. If we choose a more modern language, designed with SP in mind, such as STRUCTURED FORTRAN, ALGOL or PASCAL, then we have to face the fact that currently these languages are by no means universally available. (One could almost say that they are rare animals in the personal computer world.)

Fortunately, we can have our cake and eat it! We can write our program in a structural language and then convert it, in a straightforward manner to a non-structured language of our choice.

This intermediate language that doesn't necessarily exist on any computer is usually called PSEUDOCODE.

Pseudocode, like a flow diagram, makes no reference to any particular computer. In fact it is rather like a written form of a flow diagram and many experienced programmers sidestep the action of drawing a flow diagram by *writing* it in pseudocode. Each type of SP block gives rise to a type of pseudocode instruction. Hence we have four basic statements or collections of statements.

The Sequence block corresponds to any list of pseudo statements with each executed in sequence, e.g.

```
        statement 1
        statement 2
            .
            .
        statement n
```

The DO WHILE statement is usually written

```
        statement
    DO WHILE (condition)
        statement
        statement
    END DO
        statement
```

e.g.

```
        statement
            .
        A = 10
    DO WHILE (A is greater than 0)
        print A
        A = A – 1
    END DO
        statement
```

which means print 10, 9, 8, ..., etc. down to 1.

This first example brings us to one of the main advantages of pseudocode. As there is never any intention of presenting it to a computer we can mix precise, computer-like statements with vague English comments. For example the "condition" in the DO WHILE statement could be "(A is a perfect number)" or "(today is Friday)". Obviously before the pseudocode can be converted to a computer language these vague statements have to be made exact, but as any good arithmetic teacher will tell you, it's always better to leave the working out until the end.

The DO UNTIL statement is usually written:

```
        statement
    DO UNTIL (condition)
        statement
        statement
    END DO
        statement
```

e.g.

```
        A = 10
    DO UNTIL (A is less than 0)
        print A
        A = A – 1
    END DO
        statement
```

* More accurately, a computer language designed for SP should not contain an unconditional GOTO or JUMP.

which means print 10, 9, 8, ..., etc. down to 0.
The IF is usually written

```
        statement
    IF (condition) THEN
                Sequence block 1
            ELSE
                Sequence block 2
    END IF
```

e.g.

```
        statement
IF (today is Saturday) THEN
                do not go to work
            ELSE
                go to work at 9 am
    END IF
```

These statements should not cause any difficulty for those familiar with any existing computer language, the only unusual feature being the END IF and END DO parts. These are the exact equivalents of the END boxes in the flow diagrams. One convention worth mentioning is the use of the keyword that starts the block (e.g. IF) written backwards to end it (i.e. FI). Thus

```
IF (today is Saturday THEN
                ....
            ELSE
                ....
FI
```

This seems to be a good idea until we get to DO UNTIL and DO WHILE which give LITNUOD and ELIHWOD, neither of which adds to program readability!

The final step is to translate the pseudocode into the "target" computer language. For most languages this can be done in a routine way using a "dictionary". For example in BASIC our pseudocode statements are translated as follows.

```
IF (condition) THEN
                statement list 1
            ELSE
                statement list 2
    END IF
```

might become

```
    IF condition THEN GOTO 20
    statement list 2
    GOTO 50
0020 statement list 1
0050 rest of program
```

The DO WHILE and DO UNTIL are just as easy.

```
    DO WHILE (condition)
    statement list
    END DO
```

might become

```
0040 IF not condition THEN 55
     statement list
     GOTO 40
0055 rest of program

     DO UNTIL (conditions)
     statement list
     END DO
```

might become

```
0040 statement list
     IF not condition THEN 40
     rest of program
```

The only point we have to watch is that the "conditions" are expressible in BASIC, i.e. we have to get rid of things like (today is Friday). Apart from this the entire translation could be automated and in fact this is what is often done to modernise outdated languages.

For example, STRUCTURED FORTRAN is available as a translator (to standard FORTRAN) thus giving us the benefit of both worlds: the power of a structured language and the availability of standard FORTRAN compilers. The reader may be tempted to rush off and start writing a translator from STRUCTURED BASIC to standard BASIC but be warned! Most BASIC systems use an interpreter rather than a compiler. (The technical distinction between these methods need not concern us – it is their outward appearance that is important.) The real power of an interpreted BASIC is the level of interaction it provides. It is possible to write a program, run it, stop it, examine the contents of any variables, change them and continue running the program, etc. This is probably the fastest way to debug a program. Now, if we were to use a translator, this would be a barrier between the programmer and the interpreter. The BASIC program would be quite strange to the programmer, the variable names would have been assigned by the translator and the structure would not be the one the programmer produced. Thus interaction would be much more difficult and changing the program would mean going back to the structured source and then retranslating it. No . . . when you have an interpreter for BASIC, an automatic translator doesn't seem to be worth it. (I hope one or two of you will prove me wrong!)

It should be noted that it is just as easy to translate pseudocode to Assembler as to BASIC, proving that it is possible to write structured machine code!

Another Block

At this point we will introduce a fifth, very useful, commonly used block, the DO CASE. As mentioned earlier, which blocks are used is not fixed by theory, but it is important that we agree on a *common* set of blocks. The DO CASE has met with acceptance because it saves writing a lot of IF statements.

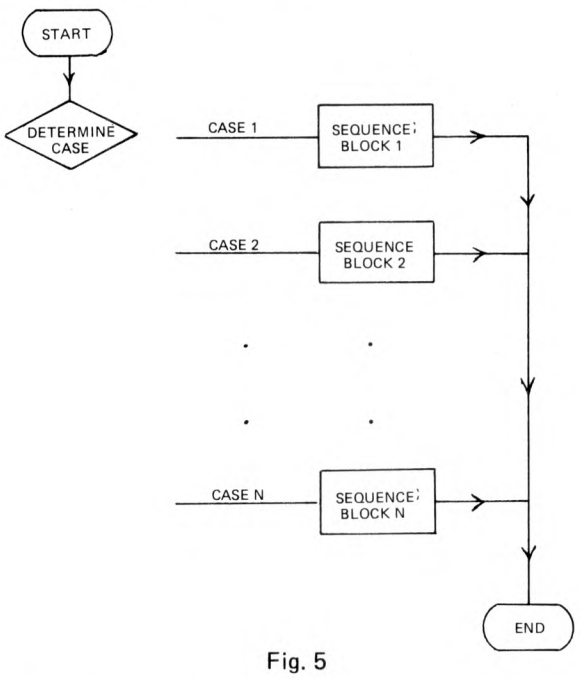


Fig. 5

The flow diagram in Fig. 5 should explain its action. One of the N sequence blocks is executed. There are a number of ways to determine which, from the most comprehensive "do case one if today is Friday", to the more restrained "IF I = 1 THEN do case two". As the latter method is closer to current programming languages this is the one we will use in the pseudo language.

```
DO CASE I
    CASE 1      sequence block
    CASE 2      sequence block

    .           .       .

    CASE N      sequence block
END CASE
```

which means IF I = 1 THEN GOTO CASE ONE etc. This leaves us with a problem about what to do if I is zero,

negative or greater than N. While the choice is somewhat arbitrary, it is usual and logical for none of the cases to be carried out and for an error to be reported in all these conditions. The BASIC translation of the DO CASE block is particularly easy. The previous pseudocode becomes

```
                ON I GOTO 1000,2000,. . .,N000
          1000  case one
                GOTO N000 + 1000
          2000  case two
                GOTO N000 + 1000


N000 +  1000  rest of program
```

Of course, the line numbers can be changed to anything that works!

```
DIM A(N)
   .
   .
   .
DO UNTIL (K = 0)
      K = 0
      I  = 1
      DO WHILE (I < N)
         IF (A(I) > A(I + 1)) THEN
                                T = A(I)
                                A(I) = A(I + 1)
                                A(I + 1) = T
                                K = 1
                                I = I + 1
                             ELSE
                                I = I + 1
         END IF
      END DO
END DO
   .
   .
   .
rest of program
```

Fig. 6

An Example

At this point an example is long overdue. Finding one that is simple enough to be understood and yet complicated enough to show some of the techniques is always a problem.

The program in Fig. 6 is a simple sorting program. Given an array A of size N it sorts the contents into ascending order using a technique known as a bubble sort. The array is scanned and adjacent elements A(I) and A(I+1) are compared. If they are in the wrong order then they swap places. The array is fully sorted when no swaps are made during a complete scan. The structured program for this process is best explained starting from the inside blocks. The IF block in the middle of the program compares the two adjacent elements of A and, if they are in the wrong order, the THEN part swaps them and increments the index I. If they are in the right order, (the ELSE part), then only the index is incremented. The DO WHILE block carries out the scan of the array A. As long as the index I is less than N then the IF block is carried out. Clearly, as each time through the IF block the index is incremented, the DO WHILE block is finally left.

(Remember that, in a DO WHILE block, the test is made *before* the sequence block is carried out.) The DO UNTIL block is responsible for the repetition of the scanning (i.e. the inner two blocks) until no swaps are made during a scan. The flag K is set equal to zero inside the DO UNTIL block and remains zero unless a swap is made by the IF block when it is set equal to 1. Thus if a swap is made, K is not 0 and the sequence block (i.e. the scan) is carried out again. The only way that K can equal zero and the DO UNTIL finish, is if no swaps are made during a scan, and, of course, this is exactly what we require. It is worth noting that, as the test for the DO UNTIL is not made until the end of the loop, there is no need to preset the value of K before entering the loop, as would be necessary with a DO WHILE loop. It is admitted that this program could be improved upon. However, it serves its purpose in illustrating some of the points, and should be studied until it is fully understood.

A BASIC translation of this program can be found in Fig. 7. Anyone who studies this program should identify the main trouble with SP — it doesn't produce the most efficient program possible. However, as this is not the intention or purpose of SP, perhaps this is not much of a criticism.

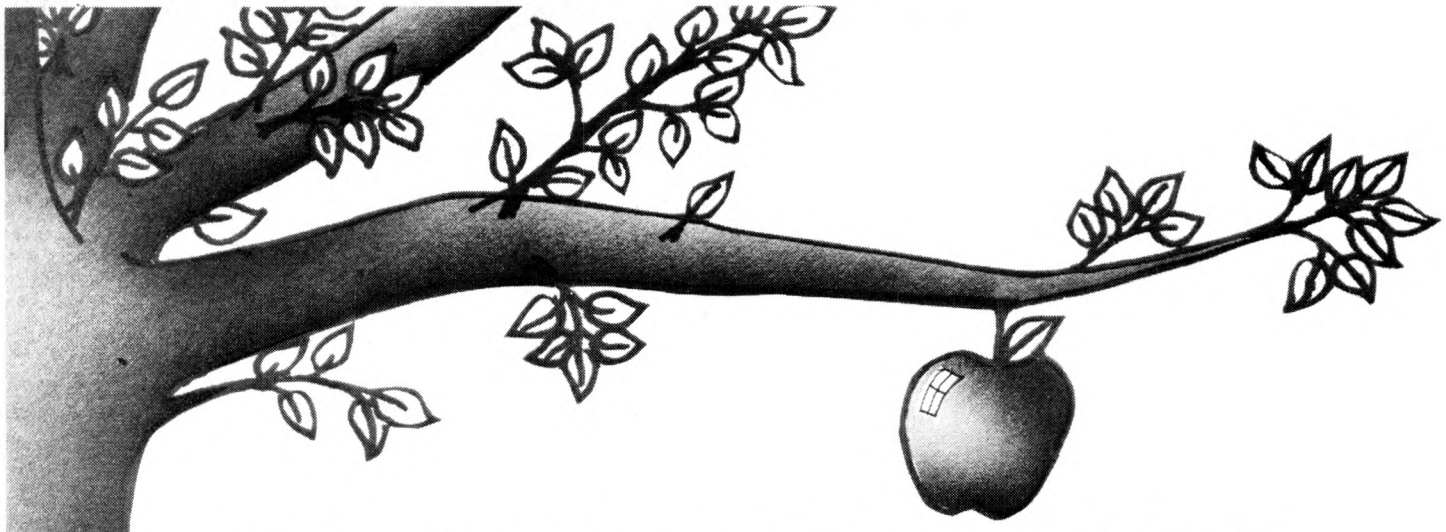
Conclusion

That's all there is to SP! If at first you don't see what all the fuss is about, then I would urge you to try it for a while — it gets easier with practice and I find that its advantages only become clear when you return to a program and try to modify it. However, home computing is supposed to be fun and I must admit that there is a lot of pleasure to be had in constructing a "clever" program and one thing structured programs are not is clever. I must admit that I sometimes sacrifice the standard structure for a clever program which gains some speed or saves storage, but it's better to *know* you're breaking the rules than proceed in a haphazard way.

In this article we have considered the subject of program structure — how we can (or should) put programs together. Nowhere in this discussion have we mentioned the concept of a subroutine, nor have we mentioned the data. In fact the title "structured programming" is usually taken to mean a bit more than we have considered. A technique of programming called the "Top Down" method and the theory of data structures are usually included with SP in any systematic approach to programming. The Top Down method is about how we (should?) use subroutines, and the theory of data structures is complementary to the theory of program structures. Only when all three are used together do we have a truly powerful method and theory of programming. The two remaining topics will be introduced in future articles.

```
                 10   DIM A (N)
                       .
                       .
                       .
                 50   K = 0
                 60   I = 1
                 70   IF I =N THEN GOTO 170
DO   DO          80   IF A(I)  A(I+1) THEN GOTO 110
UNTIL WHILE IF   90   I = I + 1
                100   GOTO 160
BLOCK BLOCK BLOCK 110   T = A(I)
                120   A(I) = A(I+1)
                130   A(I+1) = T
                140   K  =  1
                150   I = I + 1
                160   GOTO 70
                170   IF K  0 THEN GOTO 50
                180   rest of program
```

Fig. 7



Keen Computers Branches Out!

Keen Computers are now agents for **South West Technical Products (UK) Ltd** and also stock the **BUS-2**

EVERYTHING YOU NEED FOR YOUR COMPUTER!

A comprehensive range of computer stationery and accessories is also readily available:-

Floppy disk files.....£9.98
Floppy disks £3.00
Suspension binders...from £1.28
C15 cassettes £0.85

Keen Computers is a main dealer for the **Apple II** computer.
Simply a better and more advanced machine for many business purposes

* It uses a 6502 microprocessor – a very updated version of Motorola 6800
* It is a completely built system with colour graphics and comprehensive basic

Apple II Computer 16K	£1050
Additional memory per 16K	£250
Disk with controller	£425
Disk without controller	£375
Applesoft card	£100

Printers

Data 100 fast matrix printer	£1750
DDT DECwriter II	£1050
Centronics 779	£699
Centronics Microprinter	£350

SOFTWARE

As well as our normal consultancy service we also market the following software packages:-

Addressing and mailing program .. £50
Word processor upper and lowercase £50
Personal accounting £10
Depreciation calcs £10
Incomplete record accounting .. £250
Co resident assembler..... £20
Sales growth analysis..... £10
A variety of statistical packages from £25

A large range of games are available
(Prices subject to change without notice)
For a full catalogue please contact:

Dr. T. KEEN, KEEN COMPUTERS LTD., 58 CASTLE BOULEVARD
NOTTINGHAM TEL: 45865/49588

THE SORCERER HAS ARRIVED



Introducing the personal computer you've waited for. The Exidy Sorcerer.

I didn't buy my personal computer until I found the one that had all the features I was looking for.
The Exidy Sorcerer does everything I wanted to do and a few things I never dreamed of.
It isn't magic. Exidy started with the best features of other computers, added some tricks of their own, and put it all together with more flexibility than ever before available. Presto! My reasons for waiting just disappeared.

I wanted pre-packed programs. Software on inexpensive cassette tapes for the Sorcerer is available from Exidy and many other software makers.

I wanted user programmability The Sorcerer's unique plug-in ROM PAC™ Cartridges contain programming languages such as Standard (Altair 8k*) BASIC, Assembler and Editor (so I can develop system software), operating systems such as DOS (so I can also use FORTRAN and COBOL) and applications packages such as Word Processor.

*Altair is a trademark of Pertec Computer Corp.

I wanted Graphics, and the Sorcerer is super. Its 256 character set – more than any other personal computer – includes 128 graphic symbols that I can define.

I wanted high resolution video. With 122,880 points in a 512 x 240 format, I get the most detailed illustrations.

I wanted to display more information. The Sorcerer displays 1920 characters in 30 lines of 64 characters – equal to a double-spaced typed page.

I wanted a full, professional keyboard. The Sorcerer's 79-key data processing keyboard provides designated graphics, the complete ASCII character set in upper and lower case, and a 16-key numeric pad.

I wanted memory. The 12k of ROM holds a Power-On Monitor and Standard BASIC; 32k of RAM is supplied on board.

I wanted expandability. Serial and parallel I/Os are built in, and the optional 6-slot S-100 expansion unit lets my system grow.

I wanted a computer that's easy enough for children to use. I just connect my Sorcerer to a video display and a cassette tape recorder, and if I have any questions the easy-to-understand Operation and BASIC Programming manuals have the answers.

I wanted to buy from an experienced Manufacturer. In five years Exidy has become the third largest producer of microprocessor-based video arcade games.

I wanted to spend less than £1,000. (This is where COMP. does a little magic). My Sorcerer cost me £950!.

Now, what are **you** waiting for?
Call COMP. on 01-441 2922 or write to

COMP COMPUTER COMPONENTS
14 Station Road,
New Barnet, Herts. EN5 1QW.

(Price shown ex. VAT)

DEALER ENQUIRIES INVITED, A LIMITED NUMBER OF DEALERSHIPS STILL AVAILABLE.

Crofton 6800 Microcomputer with FREE Tiny Basic

The Crofton 6800 Micro. we believe offers the finest value for money available anywhere. Our kit is fully guaranteed and what's more we will even build it for you at the price shown below. Only the best quality components are supplied and all pcb's are made in our own factory.

The basic micro which is fully expandable comprises one Mother board, incorporating serial clock (TTL) and Master Clock, having positions for three of this family of boards, position for the power supply components and video character generator circuitry to produce Iv p-p composite video in to 75 ohms. Display would consist of 16 lines of 64 characters per line. The terminal is semi intelligent utilizing an LSI CRT control chip allowing ten different cursor commands.

The first of these standard boards in the Central Processor Board, which has fitted:- a 6800 Microprocessor, a Parallel to serial conversion communication port (ACIA), a 512 Word Read Only Memory to supervise the system, a 256 Word Random Access Memory for temporary storage in connection with the supervisory R.O.M. and Microprocessor shuffling of data – hence this R.A.M. is called the 'Stack' and lastly there is some sundry logic and buffering to provide control signals so that all the devices in the system talk to each other in the correct order.

The second of these standard boards is 2K + 2K memory board which comprises:- 2048 words of Random Access Memory for the user to write his programmes within, 2048 words of Read Only Memory – here the user has two choices, a) to write his own programme in the adjacent R.A.M. – debug it – then to utilize the on board programmer (a temporary Power Supply of +25V DC is required) to 'burn' the user Programme permanently into Read Only Memory for future repetitive use (which is now referred to as 'non volatile') or b) to have a pre-programmed R.O.M., which Crofton will supply ready to use, containing our own "2K Tiny Basic Interpreter © " at no extra cost to the user; again this board has sundry control logic and buffering fitted.

The third position on the Mother board can be utilized for memory expansion with one of our 4096 word R.A.M. boards, one of our 16384 word R.A.M. boards, or one of our peripheral Input/Output boards; for more details of these please call our Sales Office.

An uncased standard "QWERTY" ASCii keyboard is supplied ready built and tested.

KIT PRICE

Building and Testing Fee £40

£220·00
+ £2·00 p&p and VAT at 8%

MAKING IT WORK - The user has to have available, or can obtain from Crofton, a simple power supply capable of delivering +5 Volts at 2 Amps and + and –12 Volts at ½ Amp each.

The free Tiny Basic is a conversational subset of standard Dartmouth Basic:-

Commands:– LET IF . . . THEN INPUT PRINT RUN GOTO REM END GOSUB RETURN LIST CLEAR

Expressions:- + - × ÷
Handling numbers in the range ± 32767

Variables:- A to Z inclusive ± 32767

Functions:- Random Numbers and User subroutine call.

Special Functions:- Two subroutines have been included to enable User to put/pull data To/From a memory location.

Relational:- =, > < or < >, <, >, < =, > =.

If Basic is to be re-entered without clearing an existing programme this can be achieved utilising a second entry point.

12" Built and tested video monitor uncased £60 Power supply components (For Mother Board) £20
Cassette interface board (Kansas Std.) Kit £40
All Prices Ex-Works but + VAT (8%)

CROFTON ELECTRONICS LIMITED

35 GROSVENOR ROAD, TWICKENHAM, MIDDLESEX, TW1 4AD
Telephone: 01-891 1923