

DRAGON USER



The independent Dragon magazine

August 1986

Contents



Editorial

Letters

Two new User Groups — a routine to change the cursor shape — questions about the Dragon Plus add-on — Back issues.

People's Chart

Vote for your five favourite Dragon programs, dream up an entertaining program, and win £25 in software.

News

Winter Eclipse-Fanner? — new games software on the way — 6809 Christmas Show details.

Communication

Send in your questions, if we can't answer them, maybe we can find someone who can.

Dragonsoft

Twoyoung are (well, fairly) — Ruby Robbs and Superboy! — and a golden older, Laser Zone. We know Doy Orbsam has some weird habits, but is he getting in too deep?

Show Report

Roy Coates, who should be at home writing some more games, reports from the John Parn Dragon show at Chiswell Town Hall, where he saw some new software.

Machine Code Tutor

After a month's absence, Meers, Orbsam and Campbell proceed with flags, branches and a pot-pourri of further instructions.

Dragon dialects

Beginning a new series on language alternatives to Dragon basic, Brian Cadge looks into Pascal.

Dragon Answers

We have the answers — on data graphics storage, disk interfacing, and DelatDOS, — but what are the questions?

Screen Designer

Use all the Dragon's graphics and text facilities to design, save and display custom screens.

Sound Ability

A set of routines to make the most of the Dragon's sound capabilities — which are larger than you might think.

Sliding Graphics

Pam D'Arcy uses her basic to write a non-arcade graphics game, and explains the techniques as she goes.

Arcade Arena

Not only a map of The Dark Pit but a listing to save Total Eclipse game. We haven't tested it. This is live polishing!

Adventure Trail

Mike Gernard with problems and solutions, and a closer look at Aquanaut 471.

Competition

Lots of little puzzles this month but you need only solve one of them to win the prize.

11

SUBSCRIPTIONS are still pouring into Little Newport Street, to the delight of everyone except poor Anne Marie, who has to enter all the details. Suggestions and opinions have also been pouring in. A few people who are short of money are worried about losing touch. Help your fellow Dragon-users to stay in touch by carrying the latest issue in your back pocket and whipping it out wherever Dragons gather — back issues will be available, and new subscribers are welcome at any time.

The special offer of £12 for a year's subscription continues this month, so if you know someone who missed the June issue — draw it to their attention.

This month DU begins a new series from technical maestro Brian Cadge on language alternatives to Basic, starting with Pascal, and we re-join Orbsam and Campbell in their epic trip into machine code, with a double helping to make up for last month's dearth.

We depend on your feedback, so write and tell us what would be useful.

Telephone number
(All departments)
01-437 4343

Editor
HELEN ARMSTRONG

Production Editor
BARBARA HAJEK

Associate Editor
JOHN COOK

Editorial Secretary
ANNE MARIE O'DWYER

Advertisement Manager
SIMON LANGSTON

Administration
GERALDINE SMYTH

Managing Editor
PETER WOLKOFF

Publishing Director
JENNY INELAND

Subscriptions UK (£14 for 12 issues)
Overseas (surface) £26 for 12 issues
ISSN 0268-0177. Issue 28/86/75
Dragon User, 13/13 Little Newport Street,
London WC2H 7TP
US address: c/o Business Press
International, 225 East 42nd St, New York,
NY 10017
Published by Sunrise Books, Sun Press
Ltd. (Sunrise Books 1986)
Typesetting by Clarendon Press, Clarendon
Books. Printed by Haddon Bros, Aylesford, Kent.
Registered at the Post Office as a newspaper.
Dragon and its logo are trademarks of
Bamford Ltd.

How to submit articles

The quality of the material we can publish in Dragon User each month will, as every great editor depends on the quality of the discoveries that you can make with your Dragon. The Dragon computer was launched on to the market with a powerful version of Basic, but with very poor documentation.

Articles which are submitted to Dragon User for publication should not be more than 3000 words long. All submissions should be typed. Please leave wide margins and a double space between each line. Paragraphs should, wherever possible, be computer printed on plain white paper and be accompanied by a tape of the program.

We cannot guarantee to return every submitted article or program, so please keep a copy if you want to have your program returned; you must include a stamped, addressed envelope.

Letters

This is the chance to air your views — send your tips, compliments and complaints to Letters Page, Dragon Gear, 12-13 Little Newport Street, London WC2H 7PP.

More to come

I WAS disappointed to hear that the mag is going subscription only, but I understand your reasoning, and it's better to pay £12 in one go than not to buy GU at all!

I hope that your promise to 'back even more in' means more pages.

Would you please give our following to articles on the following: Use of Diagnostics (the manual is just about useless); adding a second disk drive using one of the many cheap drives now available; CDSPI Drive (especially an article by Computerwise) identifying (to robots etc) and more technical details on the hardware and circuitry.

D. J. Barkham
649 Obelisk Row
New Broughton Green
Northampton NN2 2TG
PS Any idea where I can get hold of a copy of the D59 operating system? I'm getting desperate.

We would like some more news on the D59 system ourselves. Can anyone assist?

Useful programs

OVER the years I have developed a number of small utility programs which I have found very useful. They include an easy to use load and search program, a program to auto-run and partially protect both Basic and M/C programs — this includes a load screen designer, a program to enable you to make copies of your own author programs, a Merge program for easy merging of Basic programs, a List program which by listing online at a time and providing easy movement up and down etc, make listing easy, and a full 6809 disassembler for the Dragon.

If any of your readers would like a copy of these programs plus instruction sheets I would be glad to send them one. They should either send a tape, size and 17p stamp to

cover photocopying or their address plus £1 to cover anything.

Poly Sewell
44 Salford Road
Widmer
Dorset
Bournemouth

Add on Colour

AFTER reading the review of the Dragon Plus in the January 58 GU I had the impression that unless one has a disk drive and wants to use Plus this expansion is of no use. Is this correct? In relation to graphics, does anyone make an add on that will give colour in PMode or? The Dragon Plus gives 80x24 display, what effect does this have on the graphics?

J. E. Smith
35 Bewick Crescent
Newton Aycliffe
Co Durham DL6 5LJ

Computers are the agents for Plus only, so they don't sell OSs. They don't do cassette-based systems either, so their own software is written around the requirements of Plus; there is no software on the board itself — if you can obtain suitable software control, you can hook it up to any system. The Dragon Plus does not alter the graphics capabilities, being a text-only display.

Well Done!

I AM writing to GU to say thank you to all the staff who helped get many a delayed copy of Total Eclipse. Without their support many readers might not have known how to contact. I hope you print this, so that people can see, that without GU, the Dragon just could not survive.

S. Moorfield
14 Clarendon Road
Great Blunston
Huntingdon
Cambs PE17 5AL

New Shape

IN RESPONSE to L.M. Mendenhall's query as to whether or not the Dragon 50's cursor could be made to change its shape, I have written this small routine to do just that:
10 FOR X=54022 TO 54015
20 READ AS
30 POKE VAL("M"+AS)
40 NEXT X
50 DATA 88, 82, 86, 87, 91, 88, 88, 78, 87, 87, 84, 79, 74, 72, 90, 86, 88, 74, 87, 84, 78, 82
60 GOTO 110

(Enter the program and RUN it. As it stands, the program changes the cursor to an asterisk, but can be altered by typing: POKE 54011,ASC ("your character").

Stuart James
24 Gainsborough Drive
Penton
Wolverhampton
WV6 7ND

Dragon Society

I WOULD like to let Dragon User readers know about a user's group I have started. It is called the Dragon and Co/Co Users Society. Members will receive a quarterly newsletter that offers hints and tips, answers to members' queries, program listings and a chance for members to get in touch with each other. Anyone joining will receive a machine code utility, to auto-run their own programs.

Membership is £1.80 a year and further information can be obtained from me at the address given.

Kevin Coleman
804 Evers Vale Road
Gower
Kent CT17 3JN

Correct Tips

IN THE April edition under "More Tips" there were two mistakes. The first line should

read:
25 FOR A=1 TO 10
The second was line 48 and should read:
40 SWRITE 1, 20, A, AS, BS
R. Baker (GATTT)
52 Princess St
Chesham Terrace
Staffs W20 6JN

Amateur Radio

I AM writing to advise you that I am engineering a Dragon Amateur Radio User Group. This group will cater for licensed and listener radio amateurs and will explore the capabilities of the Dragon (both 32 and 64) in the field of amateur radio to the full. Having seen the letters following Martin Worscott's letter, I would like to say that this group will be specialised and devoted solely to amateur radio with the Dragon. I would be grateful if you would publish this short letter and anyone interested should contact me at the address below. We have already published two newsletters and the third is due out in July. The subscription is £3 per annum.

Roger Moore GVR6AK
20 Moor-ay-ajce
Vordale
Bristol
Polytechnic
8th Floor
CT38 2JN

Back Copies

WHERE and how can I buy older back issues of GU? I mean from 1984-1985.

Lots of people

We normally only have back numbers for the last six months, although if you're lucky you can find a few older issues. We ask £1.25 post paid per issue.

We can supply photocopied articles from some earlier issues for £1 per article, irrespective of length.

If we can't supply the issue or article you ask for, your order and money will be returned.

News desk

If you have any new products for the Dragon — software or hardware — ring the News Desk on 01-637 6342

Sun starts to shine for Eclipse?

TOTAL ECLIPSE, the vast space trader game which ran into problems on its maiden voyage, looks as though it's coming out from behind the cloud at last.

Eclipse-Peripherals chief Trevor Davies believes that every customer who contacted the company has now been sent the much 1.3 version of the game, free of the elusive bugs which had stopped play. "To the best of my knowledge, there's nobody out there who hasn't received a game, or ordered one. The Birmingham

Consumer Services Department put one more chip through to me this week."

"We were quite surprised at the public's reaction. They have stuck with us. People who are really unhappy at the time have contacted us to say how pleased they are."

"It has taken time to deal with mail because we had to tell some of our staff to go, and the office is not running full time. We also lost some mail because of this. It has really knocked us for six. I put money into it and went to over the top."

"People have asked us about our real game, like are we contemplating another game, but I will take it a while to get back on our feet."

Birmingham Consumer Services confirm that Eclipse apparently has it sorted out. They're not fly-by-nights, and I know they've been making efforts. They are still based at the same address."

The address, for anyone with further queries, is Eclipse-Peripherals, Suite 10, 4 Orpington Road, Birmingham B24 8HS.

Total Eclipse, topped as game of the year by DU's reviewer before the troubles, is available from Eclipse-Peripherals at £9.95, or win one in this month's Golden Line puzzle.

Prize Books in lieu

MELBOURNE HOUSE, who have been making efforts to find a few of the last remaining *Dragon* The Dragon tapes for Dragon User's outstanding January prize-winners, have contacted us with the sad news that nobody they have spoken to has a spare tape under the counter.

Melbourne House send their apologies to all concerned, and those winners who did not receive tapes will be getting books in lieu.

Text Adventure

A NEW classic-style text adventure running under the FLEX DGB is on its way. The Curse of Conan boasts 108 locations and 28-plus characters in 47% of machine code

plus 10k in the utility-oriented space on disc. The program has a large vocabulary, and FLEX commands can be used during the course of the game.

Look out for a review from Ray Coates soon. The Curse of Conan can be ordered from R. Hunter, 46 Greenhill Road, Glen, Bus, Leics LE18 3LL, for £10 post paid.

Radio Amateurs impress Blaby

BLABY had two new games with us now and another on the way, so look out for reviews next month. Temple of Doom, Boulder Crash and Trun are the names to watch out for.

"The Dragon show in Manchester was very successful for us" says John Bailey. "A comparatively small number of people came, but they're all serious Dragon people." John was impressed by some practical demonstrations of Dragon programmes being used by Radio Amateurs to send pictures by RTTY.

Dragon User would like more information on Amateur Radio software for the Dragon and Tandy micros, so if you



Temple of Doom



Boulder Crash

have anything to say on the subject, drop us a line.



68 Micro Group

ANY solitary hardened Dragon or Tandy CoCo hackers might be interested to hear of the 68 Micro Group — an established national club dedicated to users of all 68xx systems.

Membership includes a nicely put together 45-type journal (named 68 Microcom), access to a library of assorted software (much of it FLEX and C64-B) and, of course, contact with other en-

thusiastic users. Pressing the April edition of 68 Microcom showed it to be 30 pages packed with assembler and circuit diagrams, special hardware offers (now about 48 quid for a Modern 2000, direct Modern House!) — with four pages devoted to the Dragon fleet.

Although perhaps not for new beginners, this ambitious

group (they plan to hold nationwide monthly meetings, where therein the support) is certainly worth investigating. For membership details, write to Jim Turner at 63 Mills Road, London, E11.

Xmas Show

MICROGAL'S 6800 CHERRY-BAG Show will be held on 22 November at the Royal Horti-

cultural Halls, Westminster, London. For more details phone Jenny Pope on 0726 0820. The Royal Horticultural Halls are within easy walking distance of Victoria Station, British Rail, and Victoria and St James Park underground stations.

Microdeal have two new games in the pipeline: Cuthbert and the Golden Chalice, and Tanglewood. No release date yet, but we hope to be releasing these as soon as they become available.

Bowled out

Program: Superbowl
Supplier: Compimage (Name by Cable)
Price: £2.99

LO, life in the happy land across the waters had fallen upon even harder times than when last we told of it. With the general disenchantment among the little folk with the great witch Cateletta and her foiled blessing in Zak's favorite little folk had just away their Dragons and plucked a normal world in the back of their little televisions and a new crisis had befallen across the land.

For surely whilst the people discovered Channel 4 and found how very different it was and exciting with programs of alternative arts, and minority groups having their own programs (which did taste the

hearts of the hobbits in the land who up until then had been given a fairly hard Press by Melforne House), and alternative sports.

They were Basketball and American Football. But soon the crowd didn't die out for basketball, as the little folk had not the height to get the ball in the basket and so American Football became the craze.

And high in his castle among the dark mountains the evil wizard didst laugh and count his money as he did control all of the world steadily and all at once another laugh a lot more for he had finally removed all the Dragons from the land (for it he did hate computers as much as happiness).

So, as you can see, things looked pretty damn bad for the little people. But then the good witch Cateletta didst back it and released a game based upon American Football, which shows a top view of players running up and down the pitch

and is generally a great idea. Sadly though, the good witch had had much of her energy sapped by the defeat of



Zak's son by the evil wizard and so all she could produce was a game with nice smooth graphics but little else.

And so, once again the little people didst send off for it and get all their Dragons out, and feed it up, and play it for a

couple of nanoseconds, and get bored out of their wits with it, and put their Dragons away again, and throw away the game, and go back to watching alternative arts programs.

And the good witch Cateletta was terrified from the land for letting the people down again. And the Dragons didst never come out again. And again his came the evil wizard laughed that all this had gone so well for him when he'd really had nothing to do with it. And he didst come forward and crown himself king, and his name was Cliff Slewake.

And the morals of this tale are firstly that if no one gets their A into G about producing software then computers die, and secondly those who bring out too many games in a row can't complain about too cynical reviews.

Jason Chisum



Old zoner

Program: Laser Zone
Supplier: Microdeal
Price: (See text)

A FEW YEARS ago, when I was very young, I went round to my mother's house to see a couple of new games he had acquired for his VIC-30. One of them was Laser Zone. My played it all evening and I spent hours in incredible envy of this game.

I can't remember why I loved this game so passionately, but it's still a good game, only as down by the graphics on the Dragon. I put you on two axes of a grid with objects coming at the two axes. On each axis you have a gun and the idea is to blast anything that moves out of the sky (or grid). Up and down on the joystick controls the gun on the vertical axis, and left and right correspondingly move the gun on the horizontal axis. And that's about it, save a couple of other nifty touches that take this game from the mediocrity.

The first is that by using diagonal movement it is possible to pick an alien that has landed on a certain axis off with the other gun (does that make

sense?) This is a very difficult manoeuvre to execute involving a lot of practice to prevent suicide!

The second is that it is possible to play with two people in co-operation, one on each axis, and it's here that the game comes into its own.

The sound's great, the graphics are all right (although

I prefer black and white) and it plays fine. I have given it a rating of three but I don't know whether I found the game a lot more because it had not translated well to the Dragon, or because after a very long wait I had built up my expectations too highly, so it may be worth four.

The game is well worth a

look if you can pick it up cheap somewhere (it's often going cheap at the shops) and you may find the initial pressure for it that I shared. But, be warned, this is not an easy game!

Jason Chisum



Hooked

Program: Ruby Rabbit
Supplier: Blaxy
Price: £1.99

IT HAD BEEN a bad day, outside the rain was pouring like it would never stop. Dad knows how long I'd been on this case. I thought it would never end. Armed guards, snakes, and rules. They all swirl round my head like it was some sort of aquarium.

I threw on my coat, missed, tried again, missed again, and then put it on properly. I'd never managed to learn that trick, I sighed.

She said there when I opened the door. A real picture of beauty. "You've got to help me!" she said. "I can't stop playing Ruby Rabbit." I sighed. I knew the symptoms, my God, I had them too.

"You want a drink?"

No reply.

"I'll get you a drink," I said. I sighed. I seemed to be doing a lot of that recently. I fixed her a gin. But when I got back into the room I found that all that was left was a faint memory of her perfume and a cassette in my hand. I read it. "The object of this game is to steal the precious Ruby by breaking the complex defense system guarding it."

There they were, the same old instructions. I went to the cupboard and opened it. Out fell thirty thousand Ruby Rabbit cassette tapes. There were thirty thousand others out there who were addicted.

Then I found a hint. I decided to follow it. It led to a plug stuck in a socket on the wall. Delicious. And typical of the enemy. Blaxy. That one word struck so much terror into the hearts of so many people. Blaxy. For some it meant

sheep games. But for most it meant addiction. I decided to follow the last the other way. . . . It led to the cassette recorder connected to my Dragon.

I sighed and looked up the game. It looked fine with no problems. The graphics were smooth and flicker free. The sound was great. I knew there was a connection between this and the Ruby Rabbit case that had so nearly cost me my life. I had to play. I sighed. I sighed. I sighed again for good.

I sighed. Maybe the case was unsolvable. Maybe the game was too addictive. I looked at the screen. The blue light on the portable was all that in the apartment. I sighed, and settled down to another night's play.

Jason Chisum



What a wonderful show!

Roy Coates at Oseff.

THE SECOND of the John Peen Dragon shows was held at Oseff Town Hall last weekend the 31st May. At first thought this seemed to be an odd choice of venue although after consulting a map it proved to be well thought out as easy access is gained from both the M62 and M1 motorways. This was borne out by the fact that there were actually people waiting outside before the show started.

It was nice to see a few new faces as well as most of the regular suppliers of Dragon hardware and software in attendance with some marvellous bargains and a very large selection of both programs and accessories for the Dragon.

Firstly (and deservedly so), John Peen had a large and well manned stand offering unbelievable discounts on a comprehensive range of software which included both games and utilities on both cassette and disc. Compuserve as always, were displaying their extensive range of both hardware and software, most of which centres around the RLEK and OS-8 operating systems, which, judging by the amount of people gathered around their stand was generating some real interest. Ted Ojorthal from Compuserve was keen to stress that they are still very much dedicated to the Dragon and will be continuing to support the Dragon users as long as they exist.

Blaby (as usual) were very busy demonstrating their vast range of games software with two new releases on show, Boulder Dash and The Temple of Doom with the promise of more new titles to be released shortly.

Eclipse-Farmer were again displaying their Total Eclipse program and now seem to have come through their initial bad publicity problems simply through having such a very high quality game. For those of you that have been brave enough to tackle Total Eclipse, I have been informed that there is a third universe to be released shortly which must make this game possibly the longest playing adventure ever written for the Dragon. Good old Peaksoft were again there to tempt us with a huge range

of accessories on show including everything from replacement power supplies through disc drives to sweat shirts. Although they do not supply software for the Dragon they probably have the largest stock of accessories for the Dragon.

Computas, which is one Dragon company that seems to be rapidly expanding, has a most comprehensive range of software at ridiculously good prices and a running battle seemed to be constantly in progress to get near enough to their stand to buy something! I must confess that they had

units as well as other software of specific interest to the amateur historian.

Microvision had Beanzister on display as well as representing Incentive and Software Projects by displaying Moon Cresta and Jet Set Willy.

What made this show different from previous ones were the stands that weren't actually selling anything but which were demonstrating how they used the Dragon for a specific application. Three radio amateurs (G0A12, G4TOR and G6ZG) had set up two Dragons at opposite ends of the demonstration hall and

from St Albans came to demonstrate how they have been using Dragons to assist in the excavating of a Roman cemetery in St Albans, the software that they have developed is very impressive and allows the accurate mapping of the graves to be excavated and also the comparison of various sites to be made by either displaying multiple maps on the graphics screen or by overlaying one on top of the other. Similar programs have also been written to allow for the precise measurement and comparison of the ceramic pots which are common to this type of site.

One of the stands had been taken by a young programmer who was exhibiting his software in the hope that it may be taken up by a software house. One of the packages that I looked at was the Composer Companion which co-reads with Microcalc's Composer package and allows musical data entry to be made via a musical slave as opposed to those awful DATA statements that make data entry and correction an appalling job. If you are interested, contact Jonathan Cartledge of Stanish Software, 25 Tintern Road, Chesham, Herts, Chesham SN8 7DF.

A lot of visitors to the show were bemoaning the fact that there is very little software available on disc. Having spent a long time on audio systems it is rather galling when you find that you can't use it. Maybe we shall see more disc based software from now on?

From an exhibitor's point of view, one thing that struck me was that all the exhibitors seemed to know each other well and the atmosphere was a very friendly one as opposed to a competitive one. The relationship between them being more like a family instead of a business.

In conclusion, the show was a great success judging by the smiles, and each of the Distributors that I spoke to had the same thing to say, "What a wonderful show!". If I have missed anyone out, my apologies, it wasn't been intentional but the show was busy and it was difficult to get around everyone.



many Dragon titles for sale which I for one had never heard of before as my wallet certainly went home relieved of some of it's burden!

Amateur Computing were showcasing off their new monthly magazine for the Dragon called Dragon Monthly, Imperious title eh?, as well as their Electronic Author and Gordon Bennett programs.

Granvener Software had an impressive display of goodies on show which included the 'Ocean' range of programs which consist of assemblies, disassemblers, word processors etc, etc, as well as their products to keep the amateur radio enthusiasts happy with RTTY units, Slow scan TV and the software to support these

with communicating between them using a radio link on the 2 metre FM amateur radio band, with great success. What wasn't obvious was that the software is available to the general public, if you are interested contact Blaby's John Dallas who is himself a licensed radio amateur (G11LT). The Dragon does seem to have become almost a 'vill' machine with radio amateurs all over the country and there are many groups who are sending software via radio on a regular basis. You do not have to be licensed to receive these transmissions but you will need a receiver capable of tuning in the VHF range at about 144MHz.

The Venturian Museum

Flag And Branch

Part Five of our machine code series — Jason Orbaum waves the flags.

HELLO, yes, it's me, alone again — this month talking to you about the CC flag and Branch instructions. You'll have to forgive me if I wander off the subject, it's just I'm missing Geoff and, if he's reading this, come back. And bring the disc drive with you!

First, a big thank you to a certain Mr Martyn J. Preston who, writes to inform us all of where we can get the elusive Motorola specification sheet on the 6809. Apparently in the Hitachi Semiconductor Data Book — 4 TH 86 Microcomputer there are about 30 pages on the HD68090 and the HD6809 as well as data on the 684x and 685x microprocessors and other peripheral chips (6801, 6840, 6850, etc). There is also, according to Martyn, some information on the HD68090. Martyn got his book from Farwell Electronic Components Limited, Casar Road, Leeds LS12 2PL. The stock number for the book is 171-360 and it should cost £7.50. Thanks again for that, Martyn.

So, on to the business in hand. This month, after last month's rather simplistic article we have aimed slightly higher. If you find the information laid out in a degree or you still find it too simple to read, please write and let us know at the usual magazine address.

Below is a dissection of the CC flag into its respective bits with descriptions of those immediately relevant.

The CC Register

C : E : F : H : I : N : Z : V : O :

C: Half Carry

This bit is set (contains a value of 1) when the result of any mathematical calculation results in the fourth bit of the resulting byte being set. This will become clearer after next month's tutorial on additional instructions.

N: Negative

This is pretty obvious. The bit is set if the result of a mathematical operation should be negative.

Z: Parity

Set on equality, ie if the two elements of a CMP instruction are equal the Z bit will be set.

V: Overflow

Set if the result of an eight bit operation mathematical operation. This is the way that negative numbers are denoted in binary. Next month's rather lengthy article will explain both 2's complement and BCD notation in binary. It was decided after last month I would be better to give them a miss this month and steer off the theory back into the commands and practices.

C: Carry

Set if the result of an eight bit operation causes the need for a ninth bit, ie 11111111D + 1D = 100000000D. The result, as can be clearly seen, has nine bits. The ninth bit becomes a set carry bit in CC and the byte becomes 00.

These three, for the moment, are the important bits in the CC flag. Let us now give their relevance to the branch instructions. The branch instructions covered here are not all of those in this month's table. However, they are the only ones you will need for now. I shall use the current situation with me and Geoff as an example for illustrative purposes.

BCC: Geoff comes back if and only if it's on his terms.

BNE: Geoff comes back provided it's not on his terms.

This pair, as can be deduced, stand for Branch if Equal and Branch if Not Equal, they are used after arithmetic calculation (as are most of the branch instructions).

BLO: Branch if Lower: Geoff comes back if he agrees to drink less.

BLE: Branch if Less than or Equal to: Geoff comes back if he agrees to drink NO MORE than he did before.

BHI: Branch if Higher: Geoff comes back if he agrees to give me more spirit than before.

BHS: Branch if Higher or Same: Geoff comes back if he agrees to give me NO LESS spirit than before.

BHN: Branch Never: Geoff never comes back.

BRA: Branch Always: Geoff always comes back.

Quite quick and painful really, wasn't it? The idea is that you take this, look at the table of branch instructions, match them up, and then look at the following piece of code and work out at which jumps the code will RTZ for

the numbers given.

```
10 LDA #RM
20 CMPA #0
30 BEQ POINT1
40 CMPA #50
50 SHS POINT2
60 CMPA #32
70 BLO POINT3
80 SUGA #32
90 BEQ POINT4
100 BHN POINT5
110 RTS
120 POINT1: RTS
130 POINT2: CMPA #200
140 BLO POINT5
150 RTS
160 POINT3: RTS
170 POINT4: RTS
180 POINT5: RTS
190 POINT6: RTS
```

The MV in line 10 stands for a number given from the following list. Work out where each takes the program:

100, 32, 33, 0, 232, 233, 260, 190, 50
The answers, respectively, are the following lines:

100, 170, 110, 120, 150, 150, 150, 190

If you got the exercise correct then you can congratulate yourself on passing the most, but complex part of the course. And that really is it for this month. Next month Geoff gets back from holiday (yes, all that stuff about him leaving was a big joke, and boy, it's going to be a funny when he sees it in print) and to celebrate we'll be presenting an extra long edition. So, for those of you who like to read up ahead of us next month the following interesting and varied topics will be covered:

- (1) Assembly directives (maybe, we've been promising this one for so long now that it's almost fun to not do it this month)
- (2) The differences between LSRD and BRD and other related topics, which leads nicely into
- (3) Addressing modes
- (4) Arithmetic, complete with diagrams and tables
- (5) BCD and 2's Complement arithmetic. All this and more that you will hardly believe or understand. (Gue lots of letters!)

Branch Instructions

BCC — Branch on Carry Clear

Mnemonic: BCC & CBCC

Function: If C=0 then branch to specific point

Addressing Mode: Relative

BCS — Branch on Carry Set

Mnemonic: BCS & CBCS

Function: If C=1 then branch to a specific point

Addressing Mode: Relative

BEQ — Branch on Equal

Mnemonic: BEQ & CBEQ

Function: If Z=1 then branch to a specific point

Addressing Mode: Relative

BGE — Branch on Greater than or Equal to

Mnemonic: BGE & LBGE

Function: If (N XOR V) = 0 then branch to specified point

Addressing Mode: Relative

BGT — Branch on Greater Than

Mnemonic: BGT & LBGT

Function: If Z and (N XOR V) = 0 then branch to specified point

Addressing Mode: Relative

BHI — Branch on Higher

Mnemonic: BHI & LBHI

Function: If (C or Z) = 0 then branch to specified point

Addressing Mode: Relative

BHS — Branch on Higher or Same

Mnemonic: BHS & LBHS

Function: If C = 0 then branch to specified point

Addressing Mode: Relative

BLE — Branch on Less than or Equal to

Mnemonic: BLE & LBLE

Function: If Z or (N XOR V) = 1 then branch to specified point

Addressing Mode: Relative

BLO — Branch on Lower

Mnemonic: BLO & LBLO

Function: If C = 1 then branch to specified point

Addressing Mode: Relative

BLS — Branch on Lower or Same

Mnemonic: BLS & LBSL

Function: If (C or Z) = 1 then branch to specified point

Addressing Mode: Relative

BLT — Branch on Less Than

Mnemonic: BLT & LBLT

Function: If (N XOR V) = 1 then branch to specified point

Addressing Mode: Relative

BMI — Branch on Minus

Mnemonic: BMI & LBMI

Function: If N = 1 then branch to specified point

Addressing Mode: Relative

BNE — Branch on Not Equal

Mnemonic: BNE & LBNE

Function: If Z = 0 then branch to specified point

BPL — Branch on Plus

Mnemonic: BPL & LBPL

Function: If N = 0 then branch to specified point

Addressing Mode: Relative

BRA — Branch Always

Mnemonic: BRA & LBRA

Function: Branch to specified point

Addressing Mode: Relative

BRN — Branch Never

Mnemonic: BRN & LBRN

Function: Branch nowhere ever! (This is only included for symmetry.)

Addressing Mode: Relative

BSR — Branch to SubRoutines

Mnemonic: BSR & LBSR

Function: Branch to specified address leaving present location on system stack S

Addressing Mode: Relative

BVC — Branch on Overflow Clear

Mnemonic: BVC & LBVC

Function: If V = 0 then branch to specified point

Addressing Mode: Relative

BVS — Branch on Overflow Set

Mnemonic: BVS & LBVS

Function: If V = 1 then branch to specified point

Addressing Mode: Relative

JMP — Jump

Mnemonic: JMP

Function: Jump to specified point

Addressing Modes: Extended

Direct

Indexed

JSR — Jump to SubRoutine

Mnemonic: JSR

Function: Jump to subroutine at specified point

leaving current address on system stack S

Addressing Modes: Extended

Direct

Indexed

Addressing Modes

Part six follows fast, with Geoff Campbell on the spot.

JASCH presented a piece of prose designed to explain the intricacies of the various Gforth instructions, and I for one think it was about as clear as mud. But it should become clear in time. As the brew man wrote that entire section all on his own, I thought it was time we had a column devoted to doing what we set out to do — is to teach others to program a computer. To this end, I have chained him up outside and taken over. I will cover a few subjects related to the branch instruction, some more on computer arithmetic, and an introduction to the various addressing modes, or ways of accessing information, that the processor had.

First I suspect that was all one time regarded as the most important in computer science, that of decimal representation of numbers. It is possible, and indeed most

efficient, to work entirely in binary, but remember that other people will be using your programs when they are finished, and there are very few businessmen, shopkeepers, personnel managers, (or, in short, people) who are fluent with the binary number system, so any numerical results from a program must be displayed in decimal. The most efficient method of doing this depends on the application of the program. For a program with a lot of calculations and little result display, it is most efficient to hold the numbers internally as binary, and convert them to decimal when they are displayed. This is quite straightforward, and we will present a routine to do so later.

There are other cases, though, where this will not be the most efficient method, in terms of speed if not storage. If a program is

doing a lot of displaying of decimal numbers, but very few calculations, as is the vast number of databases currently in use, it may be easier to store the numbers as decimal. Yes, I know we are not supposed to be able to do that in a binary computer, but this is where we start to cheat slightly (but only slightly), and introduce a new concept called Binary Code Decimal, or BCD for the kids out among us. BCD is very simple in principle, and is in fact not dissimilar to how in practice.

Binary code decimal

Just as, in hex, each nibble represents one digit between 0 and F, in BCD each nibble represents a digit between 0 and 9, it is therefore fairly easy to see that a single byte is limited to a maximum value of 99.

This is no problem, because we can easily string together as many bytes as we like. When it comes to displaying the number, it is fairly simple to mask off the relevant nibble, and display an ASCII character. We will be presenting an article in the not too far distant future with a collection of small but useful routines, and this will hopefully include a test of BCD arithmetic subroutines (if I get round to writing them . . .).

There is a slight variation on the BCD principle which I have never seen anyone else use, which is to have just one digit per byte. While this takes up twice the storage, it does allow for very fast addition, subtraction, and display, and the digits can be held as ASCII (00H to 99H) rather than straight binary, allowing the display routine to simply copy them directly to the screen. This is very useful for applications like game score displays, although not so good for general calculations.

Another problem with conversion from decimal to binary is that of negative numbers. Negative numbers are often taken for granted, but they are in fact one of the most abstract concepts employed by the human race. Abstract concepts and computers normally mix like oil and water, but for once there is a reasonably easy solution to the problem.

If we consider a single word within the computer (although the principle applies across the board), we would normally expect to be able to hold a number found to 65535 (a number that will fairly soon be displaced on your memory).

Negative numbers

If, on the other hand, we sacrifice the most significant bit (the left-hand one) to represent either positive (set to zero) or negative (set to one), we end up with a range from 32767 (7FFF or 0111111111111111b) to -32768 (8000H or 1000000000000000b). This we have already touched on, and is fairly straightforward. What is a little less obvious is how to convert from negative decimal to binary and back again. For example, how about -50? It is actually a fairly simple process. First, convert the number to binary, ignoring the minus sign. This gives 00110111b (using a byte, too it means less typing). Next find the one's complement of this number (in other words, convert the ones to zeros and vice versa). This is easiest done in assembler by exclusive-oring with FFH, giving 11001111b. Then add one, giving, in this case, 11001110b. This, I sincerely hope, is the binary equivalent of -50. To convert back, the process is exactly opposite.

Now a quick jump back to the test of branch instructions. Normally, branches the method outlined above for negative number, using the byte following the branch as an addition to the current PC value, giving a range of 127 to -128 bytes (NB from the start of the FOLLOWING instruction). There is a special case, however, whereby the branch is prefixed by the letter L, making it a long branch. The range is now 32767 or -32768, or, in other words, the entire memory map. This applies to any

branch instruction available. Do not lose any sleep over calculating lengths of offsets for branches, because (a) it will make no difference if a long branch is used for a jump of 10 bytes, except using up an extra byte of RAM, and (b) the assembler will pick out any short branches that should be long branches. As a rule of thumb, use short branches throughout, and change any that the assembler throws out to long branches.

Now, after that nice easy start something to get you thinking (maybe . . . I was not thinking when I wrote it). Addressing modes. Just those two words have been enough to make strong men weep, although it is in fact a fairly simple subject. It will cover the basics this month, and get more complex as and when we use them in routines.

Addressing modes

The addressing mode of an instruction is used by the 6809 to determine where to pick-up or place the data for that instruction. The 6809 has a larger and more complex range of these modes than almost any other chip, bearing some of the bigger sixteen and thirty-two bit monsters. I will cover each mode and its uses, although not all instructions can be used for all addressing modes. As we cover each instruction, the range of available addressing for that instruction will be given. The addressing modes are as follows:

Inherent Not an addressing mode as such, this means that all necessary information is included within the instruction itself. This covers instructions like INCA, which adds one to (increments) the A register.

Immediate In this case, the required data is taken from the byte (or word) following the op-code. This has the advantage of speed of execution, as the source address has already been calculated, and is in the PC register. This is common to most other processors, but, as we will cover later, the 6809 is unique in allowing the programmer to access these address calculations, and use them to produce TOTALLY relocatable code and data (try that on poor 6502/5). This can be fairly complex, so we will devote an article to it. In assembler source code, immediate data is always prefixed by the # symbol, as in ADDA, #10, which adds ten to the A register.

Extended (or absolute) Possibly the most commonly used mode, this uses the word following the op-code as an address from which to gather the data (or as an address to which to write the data). In the source code, this is shown as a number with no prefix, or, more commonly, as a label. (For example, LDA \$2787 would load the A register with the value stored at address \$2787, but LDA SCORE is much clearer. SCORE having been previously defined by use of one of the assembler directives that we might cover next month.

Direct This is much the same as Extended, but uses a single byte following the op-code as the low byte of the address, and the contents of the DP register as the high byte. This is quicker to execute, and takes up a byte less storage, making it ideal for

applications where there is a lot of data in a 256k byte area. Can be tricky making sure the DP register has the right value without wasting more space than you save.

Relative Used (to the best of my knowledge) exclusively for branches, jumps and calls, relative addressing uses the contents of the byte or word following the op-code, plus the contents of the PC register, to calculate the address, which is normally then transferred back into the PC register. Again, this can be used to make code totally relocatable, but more of that later. At the source code level, this simply means that all that is specified is the target address, and the assembler will calculate the offset needed.

Indexed Again, with indexed addressing, the 6809 stands head and shoulders above a lot of other processors, in that it has two sixteen bit index registers, allowing access to the entire memory map without having to worry about base addresses. With this mode, the processor calculates the address from the word following the op-code, plus the contents of the specified index register (either X or Y). For example, LDA, 1000, X will, if the X register contains 24, load the A register with the contents of location 1024, or the first byte on the test screen. This in itself is very handy, but there's more! If the register name is preceded or followed by either one or two pluses or minuses, the processor will use either auto increment or auto decrement modes, which means that, for example, LDA, 1000,X+, will, if X contains 24 as previously, load the first byte of the test screen into A, then increment the X register, leaving it containing 25. Conversely, LDA 1000, -X will increment the X register first, loading A with the contents of the second byte on the test screen, and the X register containing 25. This is incredibly useful for accessing tables of information, clearing screens, and about a million and one other things.

Indirect When an indirect instruction is issued, the target (or source) address is the contents of the address contained in the word following the op-code. This is useful for things like tables of jump vectors. For example, a program might display a menu of options, each a number for the user, multiply it by two (as each address takes up two bytes) placing the result in the X register, and then use an instruction like CALL, (IMPACT,X). Note the combination of addressing modes here. This is actually indirect, and the combination of modes can get quite stunning, as it is also possible to use auto increment at it. Indeed on its own is not much use, although it is possible to find situations where it could be used. However, for such a situation, it is generally possible to find a more efficient method of gaining the same result.

Well, that about wraps it up. Sorry about the lack of assembler directives (again!) but they are coming soon. Next month, among the excuses for the lack of assembler directives, we (yes, we — Jason) will be working again if I can think up a five-thousand-throat will cover some more computer arithmetic.

Dragon dialects

Brian Cudge grabs his phrasebook and learns to parlez Pascal.

IN THIS new series of articles we will be taking a look at some of the various languages available to the Dragon computers, at alternatives to the built-in Basic.

This month we start off by looking at the language Pascal. For the purposes of this article I used Lucinda Pascal from Compuserve which runs under the Plus operating system. Pascal was developed as a general purpose educational language by Nik Wirth in the late 1960s. With extended Pascal to be used to teach structured systems programming and hence the basis of the language is reflected in this style of programming.

The "Basic" approach to programming — that is given a problem, sit down at the keyboard and write a program, attempts to solve the problem by a mixture of inspiration and a lot of trial and error. While this approach can work for relatively very simple problems, it is entirely inadequate for finding solutions to programs of any real complexity. What is needed is a systematic approach to the problem, breaking it down into smaller and smaller steps until each step has a straight forward programming solution. This, basically, is the theory behind "structured programming".

Before diving into the details of the language, take a look at the (very simple) complete Pascal program shown in figure 1.

Basic structure

This shows the basic structure of a Pascal program. In this example, I have shown Pascal commands in uppercase and variables in lowercase for clarity — Pascal makes no distinction between upper and lowercase (although Lucinda Pascal does have the facility to do this).

The first line in a Pascal program always gives the name of the program (second optionally followed by the files to be used — here only the default keyboard and screen are used (Input and Output). Following this come constant, type, variable, procedure and function definitions. In the example program there are no constants, procedures or functions so only the variables used need be declared after the "CONST" command. Variable declaration before use is an important element of structured programming, implicit declaration is not allowed. We shall see later that there is a lot more to Pascal variables than it at first may seem.

Programs are made up of "blocks" of code — the whole program is the outer, top level block, next come procedures and functions, followed by procedures within procedures and so on. The lowest level block is an actual series of statements enclosed between the keywords "BEGIN" and "END". Pascal does not use line

```
PROGRAM Factors (INPUT,OUTPUT);  
  
VAR number, divisor, sumodivisor: INTEGER;  
  
BEGIN  
  WRITE('Enter a number between 2 and 999: '); READ(number);  
  WRITELN('Factors of ', number, ' are:');  
  sumodivisor := 0;  
  divisor := 2;  
  WHILE divisor <= number DIV 2 DO  
    BEGIN  
      IF number MOD divisor = 0  
      THEN BEGIN  
        WRITELN(divisor, ' ', number DIV divisor);  
        sumodivisor := sumodivisor +  
          divisor;  
        divisor := divisor + 1;  
      END;  
    END;  
  WRITELN;  
  IF sumodivisor = 0  
  THEN WRITELN('sumodivisor = ', divisor, ' found')  
  ELSE WRITELN('No divisors found = ', number, ' is prime')  
  END;
```

Figure 1: A simple Pascal program.

numbers and the semicolons are only used to separate commands for the compiler (they are not the same as colons in Basic). Therefore, with an IF-THEN for example, if more than one command follows the THEN part then they must be enclosed in BEGIN-END as shown in the Factors program.

Block nesting can be taken to as high a reasonable level, in this program there are only three levels — the main program block, the WHILE loop block and the IF-THEN block. Blocks can be thought of as representing the simplification of a problem into smaller and smaller parts.

Anything to declare?

Pascal is an example of a "strongly typed" language. What this means is that all variables must be declared together with their type before being used. Operations that can be performed on one type cannot be performed on another. Special exceptions to this are the so-called "overloaded" operators (such as "+" and "=") which can operate on a variety of types (such as integer and floating point).

Basic has only two built-in types, these are numeric (floating point) and string (character). Pascal has 4 simple types built-in, these are "integer" (16-bit numbers), "real" (floating point), "char" (single sized character), and "boolean" (true or false).

Lucinda Pascal also has the additional simple types "byte" for byte integers and "aria" for a string of eight characters. For efficient programs it is obviously necessary to use the most appropriate "type" of variable, integer arithmetic is a lot faster than using floating point for example.

Define your types

As well as the simple built-in types, Pascal allows you to define your own types to a limited extent. Often it may be necessary to deal with entities in programs (such as colours for example). In Basic we might use a series of variables assigned: RED=1, YELLOW=2, GREEN=3 and so on. Then within the program we can say IF PAINT=RED THEN... This is an example

```
CONST MAXCOLORS = 255;  
      MAXORDERS = 255;  
  
TYPE name = PACKED ARRAY [1..MAXCOLORS] OF CHAR;  
      order = RECORD  
        redorder: name;  
        yellow, quantity: INTEGER;  
        price: REAL;  
        stockorder: BOOLEAN;  
      END;  
      daybook = ARRAY [1..MAXORDERS] OF order;  
  
VAR reddependencies: daybook;  
    ordfile: FILE OF order;
```

Figure 2: Pascal type definitions.

of an "enumerated" type in Pascal. The equivalent would be declared as:

```
TYPE COLOURS = (RED,  
YELLOW, GREEN);  
VAR PAINT : COLOURS;
```

The advantage in Pascal is that the variable "paint" can only take the values red, yellow and green and not just any numeric value as in Basic. Also you are prevented from performing arithmetic on enumerated types.

Record definitions

Pascal, like Basic, has multidimensional arrays of any type. Unlike Basic the array is not the only data structure available to us. One of the most powerful features of Pascal is its "record" definitions. A record is a data structure consisting of a fixed number of components of various types. For example, if we needed to deal with "customer orders" which consisted of customer name, part number, quantity, price etc. Then in Basic the only solution would be several arrays such as NAMES, PART, QUANTITY etc. In Pascal a record type could be defined followed by an array of these records. Figure 2 is an example of this type of definition, which also shows the use of constants in definitions.

"Order" is defined as a record type which consists of the elements mentioned previously. The type "Daybook" is then defined as an array of order records. Note that type definitions do not declare variables, only types of variables — it is then necessary to declare a variable such as "todayorders" of the type "daybook". "Order" declares a list type consisting of orders so that todayorders may be read and written to disk as whole records.

To access a particular field of a record we use, for example:

```
todayorders[5].partno := 88;
```

That is, the record name followed by a subscript, followed by the field name, some versions of Pascal allow the following type of command:

Figure 2: Variable scopes and bindings.

```
PROGRAM EX1;  
  
VAR main : INTEGER;  
  
PROCEDURE p (param1:INTEGER);  
VAR localp : INTEGER;  
  flag : BOOLEAN;  
  
  PROCEDURE q (main:INTEGER);  
  VAR localq : INTEGER;  
  BEGIN (* q *)  
    main:=789;  
    localq:=1;  
    localp:=10;  
  END;  
  
BEGIN (* p *)  
  q(flag);  
END;  
  
BEGIN (* main program *)  
  main:=10; printa;  
END;
```

```
WITH todayorders[5] DO BEGIN  
  partno:=88; price:=1.08 END;
```

This saves the programmer from having to type todayorders[5] before each field name of a particular record. Although this is a very useful function, Lucidata Pascal does not support it.

As mentioned earlier, Pascal is a block structured language, hence procedures may be declared and may be nested. The Basic "GOSUB" can be thought of as a very simple equivalent to the Pascal procedure. I will only deal with procedures here, but Pascal functions can be thought of as procedures which return values — procedures are called as expressions; $X:=F(X)$.

In Basic, a variable may be used at any point in the program and there is only ever one "version" of a particular variable. In Pascal, variables (or more formally "identifiers") take what's known as "scope", "binding" and "environments". A procedure may use its own variables which are not accessible by any other part of the program and whose values are not kept between calls to the procedures, these are known as "local" variables. Local variables may have the same name as variables in the main program ("global" variables) or as local variables in other procedures. Pascal allows recursion by the use of local variables — a new version of the variables is instantiated for each recursive call of the procedure.

The "scope" of a variable is that part of a program in which it may be accessed. When a variable name appears more than once, for example as a global variable and a local variable in a procedure, it is said to be "bound" to the current block. The variable exists only as long as the block in which it is declared is active. The environment of a block is the surrounding blocks in which it may access variables. The environments of all procedures include the main program (as this encloses all procedures) and also includes any procedures which enclose the procedure in the program text.

To clarify all this, take a look at Figure 3, this is a completely useless Pascal program, but demonstrates the various scopes etc. of variables.

Binding declaration

Here the variable "main" is declared as a global integer variable and as a local (scoped) variable for the procedure "Q" — it is declared as the parameter passed to "Q". Hence, when the procedure "Q" is entered, "main" receives a new "binding" to the scoped value and the global value of "main" cannot be accessed within "Q". When "Q" ends, "main" restores its binding to the global integer value.

"Local" is a local variable of the procedure "Q" and so cannot be accessed by any other part of the program. Similarly, "localp" is a local variable of procedure "P", but as procedure "Q" is environment includes procedure "P" (it is enclosed by it) it may access P's local variables. Notice that the main program cannot procedure P, but cannot call procedure Q directly if

surrounds P, but does not directly surround Q.

All this binding, environments and scopes may seem confusing at first, but with a little practice you'll wonder how you ever coped with Basic's variables.

I can only begin to touch on some of the many features of Pascal in an article of this size, there have I been room to mention the "heap" and pointer variables, but it is worth mentioning some of the particular features of Lucidata Pascal. Programs are written as standard text files onto a Flex disc, then compiled into a "P-code" binary file. P-code is an opcodes for a theoretical computer which the runtime program interprets and executes. The result is an impressively fast program which is stored more compactly on disc than stand alone machine code. My only complaint is that the compiler does not recover well from program errors when compiling; once the initial error has been reported a large number of spurious error messages may appear before the compiler gets back on the "right track". There are plenty of standard methods for error recovery in compilers (such as Turing's Algorithm) and it's a pity that Lucidata do not seem to have used one.

Lucidata Pascal

Lucidata Pascal has a number of very useful string handling procedures predefined for the programmer (strings are traditionally the most weak area of Pascal as well as some sophisticated file handling commands to access both sequential and random access files etc). A particularly very useful feature is the "overlay" procedure which allows very large programs to be run by swapping in and out blocks of code during execution.

The accompanying manual gives a summary of all the features available and plenty of detail on non-standard additions to the language. There is also a large section on the internal implementation of the software to allow you to customise the run-time system by adding new built-in procedures (commands) etc. This section is certainly not for the novice and some knowledge of compiler construction is useful here.

In all important respects Lucidata Pascal conforms to the ISO Pascal Standard and most textbook programs will run without change. A number of demonstration programs are included on the disc, these are all fairly elementary but do demonstrate some of the unique features of the implementation.

If this has whetted your appetite for more then there are plenty of books to be found on the Pascal languages. A few of the better ones to look out for are as follows: PASCAL, An Introduction to Mathematical Programming by Fredkin and Wain. PASCAL User Manual by Jensen and Wirtz. Programming in PASCAL by Peter Grogono.

System used: LUCIDATA PASCAL from Compuserve — 286.
Requires: PLUS operating System (B4F — minimum 1 Disk Drive).

If you've got a technical question write to Brian Cudge. Please do not send a SASE as Brian cannot guarantee to answer individual inquiries.

Dragon Answers

THE IMAGE of style this month is not a new format — it's thanks to the Pacific, who seems to have confused Brian Cudge's column with Brian's faithful Dragon coupled up another copy of Dragon Answers, but not copies, alas, of the original questions Brian has written a summary of the questions from memory — we hope you recognise your own problems!



Here on 8279 514895, they can supply Dragonites compatible controllers for CIO.

Delta DOS

I HAVE a Dragon and Delta DOS and have transferred much of my software to it. However, some games will not run if Delta is attached even though I can load them from disc and then relocate them. Is it possible to switch out Delta so the software has loaded from disc?

Andy Warner

IF YOU add the following lines of assembly language to your routine to relocate code it should allow most programs to run so if Delta Dos was not attached.

```
LDX #0053
STX 304
LDX #0100
STX 372
LDA #07
STA 373
```

What this does is to reset the interrupt vectors and also stops Delta from intercepting commands.

Interface

I'VE recently obtained a Tandy Color disc drive and interface. Unfortunately, the interface does not seem to work with my Dragon. Is there a simple way of making it compatible, and if not where can I get a suitable controller?

Adrian Richery

THE TANDY disc interface is certainly not compatible with the Dragon and there is no simple way to remedy this. You would have to change the ROM software to operate the cartridge with the Dragon.

However, Tandy disc drives are otherwise standard 5-inch drives and can be used with any suitable Dragon disc interface. Try contacting PNP Communica-

Extra RAM

I HAVE had a Dragon 64 for a couple of years and would like to know if it is possible to use the extra 32K of RAM to store graphics screens, when in 32K mode (with Dragonites).

Is this possible from Basic as I am a complete novice when it comes to machine code?

Gary Bell

TO USE the extra RAM for this purpose is not possible directly from Basic, however the program listed below will allow you to save and retrieve screens using the USR command from Basic.

The routine automatically looks at the current graphics mode — when it starts and how much RAM is used and transfers this to the extra RAM at the offset given in the USR command. USR is used to store the screen, USR+1 retrieves it. There's room for 5 PBOKE 3-4 screens or 10 PBOKE 1-2 screens or a mixture of both. For example, to save a screen on display, at an offset of 56, in the extra RAM use 5-USR(56*144), and then to retrieve it later use 5-USR(15144). The offset should be in the range 0 to 20480.

```
10 GOTO LOAD GRAPHICS FOR 56K
20 CLEAR 280-32768
30 FOR I=32768 TO 32768/640
  40 PBOKE I-32768: AG=POKE I,VAL I/64: AG=NEXT
  50 NEXT I: PBOKE I-32768/640: AG=I-32768
```

```
50 DATA 50, 10, 40, 40, 47, 50,
  50, 50, 57, 25, 73, 57, 57,
  50, 10, 57, 50
60 DATA 50, 50, 5, 50, 47, 40,
  50, 50, 57, 25, 57, 28, 50,
  50, 50, 50, 50
70 DATA 50, 57, 57, 50, 50, 50,
  50, 50, 57, 57, 57, 50
```

Graphics

I HAVE been mucking about with the Dragon's semi-graphics mode. Could you please tell me how I can implement these modes while using my Corona drive without corrupting the disk workspace.

J. M. Paster
47 Exington Road
Barbary
Dun
S210 (SA)

THE easiest way to set up the start address of the screen display is to use the normal Dragon Basic commands PBOKE and SCHEM, followed by poke to set up the semi-graphics mode. For example, to set up page 24 starting at the first graphics page you could use the following lines:

```
10 PBOKE 4,1: SCHEM 1,1
20 PBOKE 50000, (PBOKE 50000/255) AND 3)
```

New line

I HAVE recently obtained a printer for my Dragon computer, but when I list a program all I get is a one long string of output — how can I make the printer start a new line at the every listed line?

K. Mingo

THIS question appears more regularly than most others (it's more popular than the 'speed-up poke'), but is worth repeating

occasionally for the benefit of new readers. The solution is to type the following immediately after powering up your Dragon:

```
PBOKE 150,50 for 40 — number
chars per line)
PBOKE 320,320/PBOKE 320,3
```

If this causes a blank line between each listed line then miss out the PBOKE 320,3. If this still doesn't work then PBOKE loc 320 with your printers' code for carriage return and loc 320 with the code for linefeed.

Routines

I AM trying to add some new commands to Basic and want to make use of some of the routines in the Delta DOS ROM. Can you tell me where I can obtain details of these routines?

K. McCaffish

TO MY knowledge the routine documentation for Delta-Dos/Delta DOS has written published and I have very little information on this DOS. Perhaps one of our readers has worked through the ROM and could help Mr McCaffish?

'Windows' solution

OVER a year after the 'Dragon Windows' program was published (July 1985) I am still getting letters about it so we will take this opportunity to answer all the queries in one go!

Due to a minor bug in the listing the program may occasionally crash on some machines. The solution is to add the following line to the Basic loader program:

```
5 PBOKE 31814,542
```

'Windows' cannot be used in BASIC mode on a Dragon 64 as it uses a number of ROM calls, but will operate in 32K mode. Finally, many people have asked about using the software with Dragon-60. There is no simple way of changing the loader to operate with Dragon-60; a disk version has been written but this is a completely rewritten version and is not due to be published in Dragon Men.

Screen Designer

Dennis Riley text, colour and graphics to create designer screens.

THANKS TO ALL of the recent articles on machine coding, in particular the "Fireware" series, in *Dragon User*, I now find that programs in machine code are getting easier to write successfully. However, this left the problem of wanting programs to Auto-run, fortunately solved by Brian Cudge (*Dragon Answers* Aug '85), and to displaying a customised screen while loading. Hence Screen Designer.

Screen Designer allows a text screen to be built up using all of the graphics characters, colours and the majority of the text characters (some are used as function keys). A screen may be saved and re-loaded so that it may be re-used or modified. Text, colours and graphics characters can be edited and manipulated on screen to produce the desired effect. A program can be saved and made to Auto-run, displaying the screen created, and the program executed, in that pressing RESET will re-execute the program.

Entering the program

Two methods to enter the program are given. **Listing 1** gives a listing of the machine code which can be copied into memory, using the Basic loader provided (**Listing 2**), which will allow you to enter the code in stages. Just enter as much as you like, and **SAVE**, then an **Auto-starting** **LOADM** and continue from where you left off.

Listing 3 gives the Aldrom assembler listing. It will be noticed from this listing that the program starts with a short piece of code which relocates the program to its workspace at loc 30001, this has been included as an example, the need for which will be given later.

When the program has been successfully entered save the program using:
SAVEM "DESIGNER", A#0000, A#0001, A#0000

The program can now be run using
LOADM "DESIGNER", EXEC

Program operation

On running the program a Menu of options is given.

1) **Create Screen** This option is used to build up the required text screen and uses a number of function keys. On entering this option you will be faced with a green screen with a flashing cursor in the top left hand corner. Green is the default background colour (this is Graphics Green Code 145 and not text Green Code 325, this colour can be changed using the "B" key, by pressing the "B" key the colour will change for each press, and with the exception of the black screen a cursor will again appear, when text or a character is entered on to the screen or the cursor is moved.

The cursor can be positioned anywhere on the screen, without destroying existing text or characters, by using the Arrow keys.

Auto-repeat is incorporated into the program to facilitate movement of the cursor and entering text or characters.

Text can be entered on to the screen as normal and deleted using the **CLEAR** key, it will be noticed that both the **CLEAR** key and **SPACEBAR** take on the background colour.

Graphics Characters are placed on the screen, under the cursor, using the "a" key, repeated pressing of this key will change the character shape.

The colour of the character under the cursor can be changed using Shift "B", again repeated pressing will change the colour, and the selected colour will remain in operation until altered.

Both the character shape and the colour are stored into memory, and can be repeated any where on the screen using the "u" key, hence the auto-repeat makes light work of borders, though these are best left until last.

To help in the design of the screen there are six scrolling functions, the Shift arrow keys will scroll the whole screen in the appropriate direction and by using the "<" and ">" keys the individual line containing the cursor can be scrolled left or right.

Colour

The colour of all of the characters on screen can be altered using the "B" key, without affecting any text. However, as the colour of the space and dots will still be the original background colour, the use of this function is best left until a screen is completed.

The first function uses the "N" key and this will input the character under the cursor, to colour characters not normally available from the keyboard, i.e. inverse numerals.

The **BREAK** key will return to the Menu, and as the screen is saved to memory, obviously any previous screen will be lost.

2) **Display Screen** This option will display the screen held in memory, either one that is being worked on or one that has been loaded into memory. All of the described functions apply to this option.

3) **Save Screen** Here a screen can be saved to tape, and therefore a library of screens can be built up, or an unfinished screen can be saved for future work to be done on it.

4) **Load Screen** Loads a previously saved screen back into memory, together with the original background colour and the text character colour used, for use with the spacebar and dots.

Options 5 and 7 turn the motor on and off respectively, this is to allow for the clearing and/or positioning of tapes. The **BREAK** key used from menu will return to basic.

Auto-running

Option 5 will convert a machine code

program into one that will Auto-run and will display the screen chosen while loading. There are, however, some very important conditions which must be met for this to be successful.

The source program must have been saved using
SAVEM "TITLE", N1, N2, N3 where:

N1 is the start address of 5H0000. Programs must start at this address. Any program needing a higher workspace can be relocated using a similar method to the example given.

N2 is the end address of the program while in the graphics area.

Most importantly N3 is the address from which the source program is to be executed (**EXEC**) when it is in its required workspace, even if this means that the **EXEC** address is higher than the end address, if N3 points to some other address it may result in the program crashing, and pressing reset, instead of re-executing the program will certainly cause it to crash. Ideally the **EXEC** address (N3) will be the same as the address **JMPD** to it in the relocation program. Of course if the program is to remain in the graphics area so much the better.

An example would be if Screen Designer itself were to be saved using Screen Designer.

Firstly the program would be loaded and executed, it is now in its workspace at loc 30001. However, as Screen Designer does not use Hires, the original program is still in the Graphics area, this can now be saved using:

SAVEM "AUTODESK", A#0000, A#0001, A#0001

If the program is now saved, using Screen Designer, it will Auto-run on loading.

As can be seen, from the assembler listing, the relocation of code is very simple for anyone using the Aldrom assembler. As there are two directives **ORG** and **PUT** which do the job for them.

Anyone referring to Brian Cudge's piece on Auto-running machine code will see that he has given a little piece of code that should start any auto-run program, with Screen Designer. However, this is unnecessary as this code is already taken care of.

The first piece of code at the execution address should be a **NOP**, this is so that if the reset button is pressed the program will re-execute, if a **NOP** is not present, a cold start will occur.

Option 5 includes prompts where the motor is turned on or off to enable tapes to be correctly positioned.

Hints and tips

Most of the code used in this program was gleaned from the pages of *Dragon User* at some time or another, particularly in

regard to logic instructions. (Bruce Deane O.U. Jan 1981).

The routine called, using the "B" key, loads each character in turn from the test screen into the A accumulator; a test character is by-passed but a graphics character has its value increased by 16, to alter its colour. The A accumulator is then OR'd with #126 (0006) to ensure that a graphics and not a test character results.

In Two's complement arithmetic numbers 0-127 represent positive numbers (as 1 is not set) and numbers 128-255 represent

negative numbers (as 7 is set), also, not by coincidence, numbers 0-127 represent test characters, whereas numbers 128-255 are graphics characters. Therefore, by using TST and the conditional branches BMI and, as in this case BPL, it is possible to distinguish between graphics and test characters, and branch accordingly.

The previously described routine, if used as a subroutine, can do wonders for Mouses or any test page that is waiting for a prompt. Listing 4 gives a short example, a little

over the top, maybe, but it does demonstrate what can be done to liven up a screen.

All of the IO functions are from Brian Cudge's "Firmware Series".

Entering machine code, especially without an assembler, isn't de Mosaic, so I will provide a copy of the program for £3 sent to Dennis Riley, 31 Colmore Road, Moseley, Leeds LS2 4DP. I will answer general questions about the program, but please send a stamped, self-addressed envelope.

Listing 4: machine code test.

0600	0E75B1196C96126CA1ED	= 1129	07FE	32304FAE0C9929372E3E	= 495
060A	20E07E0F30F7FE19719E	= 1269	0800	2E2E2E2E2E2E2E2E2E2E	= 468
0614	30933333343336377A2A	= 531	0812	2E2E2E2E2E2E2E2E2E2E	= 595
061E	7A297A797A7E7A4E7E14	= 1899	081C	32304FAE0C9929372E3E	= 539
0628	7A2E7A2E1F2E2E7E3E2E	= 1879	0826	43410E2E2E2E2E2E2E2E	= 731
0632	03753A1F82E2E2E2E2E2E	= 1819	0834	2E2E2E2E2E2E2E2E2E2E	= 468
063C	1F2E2E2E2E2E2E2E2E2E	= 369	083A	2E2E2E2E2E2E2E2E2E2E	= 542
0646	9E6E8E129A3E3E3E3E3E	= 399	0844	43434343434343434343	= 709
0650	132E0C243F3E133C3C3E	= 933	084E	2E2E2E2E133493245448	= 641
065A	2E2E2779277A277C477E1	= 1236	0858	2E2E2E2E2E2E13C6353A2E	= 936
0664	7A7977D4777F787E7827	= 1231	0862	0073E17E5E5E79A2020E	= 798
066E	79A27E637E9C7E957E64	= 1329	086C	FF00FF00FF00FF00FF00	= 1314
0678	70477E32794C7E9549E9	= 919	0876	3E8E2E8E03FF132E329E	= 826
0682	FF00FF00FF00FF00FF00	= 789	0888	3F3E1443299E0E330E3E3	= 1174
068C	4E442E034F553E13A3E2E	= 679	0894	0C94901925E2E0F7E91E	= 637
0696	5A15845E3E944F2E953E4	= 683	089A	9E9E9E9E9E9E1F8C949E1E	= 671
06A8	413E9A2E841AE44E2E9E2E	= 669	089E	25927C9F2E15E27E3E2E	= 997
06B4	4E93E2E06E6E74E57E2E	= 641	08A8	39919C93FF132E3E0D9F	= 789
06B8	5A4F2E944F4A4E9E4C4F	= 639	08B2	8E1E9E9E8E6FF9C0C4EFC	= 1222
06C2	414E2E4A4E3E3E3E4E4E1	= 685	08BC	00F77542FA7E949E9E9E7	= 1769
06C6	5A494FAE2E94415E43E2E	= 676	08C6	94E0E94E277E81A827E2E	= 1249
06D0	414E4E9E9E9E9E9E9E9E	= 435	08D8	1E4E2E4CE6F77E0E9F87E	= 1358
06DC	5E924E5E3E3E3E3E3E74E5	= 937	08DA	09C819C1793273F777E09	= 1694
06E6	7E9E9E94F5E4E34E94F4E	= 743	08E4	FA734CE69E794E0E9E9E	= 1369
06F0	2E444E3E3E4E3E4E15449	= 789	08EE	27F8E113192E94237E9E9	= 985
06FA	4FAE9E94415E4E2E0414E	= 669	08F8	3E98F675A2FA7E0E9E7E4	= 1772
0704	41E0E9E2E9E3E13E3E3E2E	= 689	0902	3E913F8E81E6E2139E8E9C	= 829
070E	6E6E74E57E9E3E3E3E3E2E	= 818	090C	04901927E9E0E67E0E8E1	= 789
0718	5A4F2E924F3E34F5E4A9E	= 674	0916	7F2E9E9E8E17E8A7E99F	= 1842
0722	5A4E4E4E3E9E9E3E3E3E3E	= 722	0920	5E1E8E1FA2E8E3F77E0E8E	= 1314
072C	2E6E6E74E57E3E9E9E9E4F	= 681	092A	8A7E7A7E0E0E0E8E8E277E	= 1381
0736	5E4E1E9E4E3E9E9E9E24F47	= 679	0934	9E1E2E1E9E9E1E4FA7E8E9C1	= 1226
0740	5E414E0E840E4E3E5E0E2E	= 563	093E	FF24E3CE1E777E0E0E0E	= 1698
074A	2E812E2E2E2E2E2E2E2E2E	= 449	0948	7E9E9E4E079E2E9E4E9E9C	= 1231
0754	2E2E2E2E2E2E2E2E2E2E1E	= 559	0952	9E9E9E9E19C0CE8E2E3F61E	= 1262
075E	5A4E3E9E3E1F3E15E4E0E0	= 646	095C	9E7E0CA8E9E9E2ECA1E0E1	= 1617
0768	2E2E9E3E2E2E2E2E2E2E2E	= 436	0966	5C9E9E2E3F71E91E9E0E7E	= 939
0772	2E2E2E2E2E2E2E2E449E3E9E	= 589	0970	8E2E0E3FFA6E9E8E47E43E	= 1433
077C	4C415E9E9E3E4E3E2E434E	= 719	097A	3FC0E41E32FA1E9E7E0CA	= 992
0786	8E2E9E9E3E2E2E2E2E2E2E	= 494	0984	314E9E1E9E9E4E9E8CA1ED	= 1223
0790	2E2E2E2E2E2E2E2E2E2E2E	= 468	098E	818C842E2E3F71E91E87E0	= 936
079A	5E415E4E3E9E3E3E24E4E5	= 785	0998	79E2E8E4E0E0E1FA6E81A7	= 1888
07A4	4E9E9E9E3E4E9E2E2E2E2E	= 437	09A2	9E5A2E6F9E9E818C8E9E2E	= 797
07AE	2E2E2E2E2E2E2E2E2E2E2E	= 368	09AC	F0E9E441F18E7E0CA4E4A	= 1246
07B8	2E4C4FA3142E9E347E3E4E	= 687	09B6	A7E431A8E3E9E2E0E2E	= 919
07C2	4E4E9E9E9E9E3E2E2E2E2E	= 461	09C0	8E2E5F1E4E1E9E0E79E3E	= 1919
07CC	2E2E5E3A15E4E2E91E1E3E4	= 661	09CA	3FFFC63FA6E1FA7E43E1F	= 1864
07D6	4F2E3E2E4E2E9E9E2E4747	= 713	09CC	5A2E6F79E1F8C84E8E2E2E	= 878
07E0	5E414E0E8E8E8E2E2E2E2E	= 493	09CE	8E84E81E9E7E0CA314E1F	= 593
07EA	8E2E2E2E2E2E2E2E2E2E2E	= 368	09D8	4E44A7E43E1A8E2E3E9E2E	= 1894
07F4	8E2E2E2E2E2E2E40AF3E4F	= 595	09E2	8C8E1E2E3F1E8E1E2E8E7A	= 1818

Listing one.com

99FC 213068E406843404C61F = 1099
 9A06 A681A7009A26F93384E7 = 1127
 9A10 9416010A07A21301FE8 = 919
 9A14 903480C61FA631FA78439 = 961
 9A24 1F3A26F73504E7845000 = 848
 9A36 EF8D79628E0A04A0044D = 1248
 9A38 24029B100A88A7800C06 = 989
 9A42 0023F01640069E98E684 = 1169
 9A4C 0848E7841680CE129F6F = 999
 9A56 7F75A2800A778E040E9F = 1292
 9A58 989E76528D7A158E046E = 1068
 9A6A 9F000E76678D7A158E04 = 1137
 9A74 639F000E773F8D7A158D = 1491
 9A7E 000627F0F67331100E73 = 1111
 9A80 32A1A01027F9B0C0A06F7 = 1209
 9A92 20E3B0748C8E0400100E = 892
 9A9C 780AEC081EDA10C860023 = 1297
 9AA6 F739100E0A000E78DAEC = 1105
 9AB8 81EDA1100C060023F639 = 1079
 9ABA 3426CFFFFF04150F081 = 1392
 9AC4 52FD0154FD0156FC04150 = 1102
 9ACE 100E2320313F26FC35A6 = 834
 9AC8 34127375A50E08A68438 = 1195
 9AD2 484784000172001F26FC = 992
 9AEC 35327D73A527808D79F6 = 1204
 9AE6 303A12A0000D000C0206 = 945
 9B00 F03932A6801F0C41F26 = 1093
 9B0A F039066FF77B0C8B0A79 = 1728
 9B14 9E44009F987301A03002 = 665
 9B1E B07908D79F00000000027 = 1444
 9B20 F0810027F4100E7500F6 = 1290
 9B22 7067A1A01027FAFF5A06 = 1229
 9B2C F7BD7A0C010036000067E = 1092
 9B46 08817F042300010C800C = 981
 9B50 300C7901400D790215FE = 1199

9B5A F08D79C620048D7A0C8D = 1679
 9B64 8A778E04009F088E7781 = 1198
 9B6E B07A158E797734100E7E = 1047
 9B70 080F01E794101F10C302 = 931
 9B82 0234200E79710F08E534 = 909
 9B8C 107099100D7AC9008003 = 1306
 9B92 0000000074816F0B00 = 1237
 9BA0 00137E79900000107E73 = 1159
 9BA4 9080000701D10775A38D = 1342
 9BB4 8A770E77808E7775100E = 1004
 9BBE 01D02D7A158D79F00000 = 1416
 9BC0 0027F0010027D0C810027 = 870
 9BC2 0E00000A7A0A7097A73 = 1268
 9BC4 A327AFC000000A7A0A7 = 1135
 9BD6 C07A73A326F720078D79 = 1229
 9BF0 F0810026F939000A778E = 1369
 9BF4 75468D7A158D000150080 = 1278
 9C04 00810026F900000100D7A = 1007
 9C0E C900000000000000000748 = 1339
 9C10 BF73900E01E00F75A400 = 1049
 9C22 8A778E73D4007A158D00 = 1431
 9C2C 00810026F900000100D7A = 1142
 9C36 778E760007A158D000E = 1040
 9C40 01026F90000918800A77 = 1264
 9C4A 9E7628D7A158D000081 = 1094
 9C54 0026F90D7AC90079C68E = 1462
 9C5E 0000100E78C9A60A700 = 1106
 9C60 4026F90E78E701670003 = 1096
 9C72 008F01C88E78013A100E = 949
 9C7C 01078F01E7341000710E = 1050
 9C86 8A100E75A00F01E03419 = 1024
 9C90 7E991000390701678E84 = 1106
 9C9A 4F0F720E00000C010A7F = 957
 9CA4 002A25FB15FD0A0000007 = 1000
 9CAE 0157326D001E59F727E = 1071
 9CB8 050000000000F000F00 = 516

Listing two: Address assembler listing

0000 0000 0000 0000 0000 0000 0000 0000
 0004 0000 0000 0000 0000 0000 0000 0000
 0008 0000 0000 0000 0000 0000 0000 0000
 0012 0000 0000 0000 0000 0000 0000 0000
 0016 0000 0000 0000 0000 0000 0000 0000
 0020 0000 0000 0000 0000 0000 0000 0000
 0024 0000 0000 0000 0000 0000 0000 0000
 0028 0000 0000 0000 0000 0000 0000 0000
 0032 0000 0000 0000 0000 0000 0000 0000
 0036 0000 0000 0000 0000 0000 0000 0000
 0040 0000 0000 0000 0000 0000 0000 0000
 0044 0000 0000 0000 0000 0000 0000 0000
 0048 0000 0000 0000 0000 0000 0000 0000
 0052 0000 0000 0000 0000 0000 0000 0000
 0056 0000 0000 0000 0000 0000 0000 0000
 0060 0000 0000 0000 0000 0000 0000 0000
 0064 0000 0000 0000 0000 0000 0000 0000
 0068 0000 0000 0000 0000 0000 0000 0000
 0072 0000 0000 0000 0000 0000 0000 0000
 0076 0000 0000 0000 0000 0000 0000 0000
 0080 0000 0000 0000 0000 0000 0000 0000
 0084 0000 0000 0000 0000 0000 0000 0000
 0088 0000 0000 0000 0000 0000 0000 0000
 0092 0000 0000 0000 0000 0000 0000 0000
 0096 0000 0000 0000 0000 0000 0000 0000
 0100 0000 0000 0000 0000 0000 0000 0000
 0104 0000 0000 0000 0000 0000 0000 0000
 0108 0000 0000 0000 0000 0000 0000 0000
 0112 0000 0000 0000 0000 0000 0000 0000
 0116 0000 0000 0000 0000 0000 0000 0000
 0120 0000 0000 0000 0000 0000 0000 0000
 0124 0000 0000 0000 0000 0000 0000 0000
 0128 0000 0000 0000 0000 0000 0000 0000
 0132 0000 0000 0000 0000 0000 0000 0000
 0136 0000 0000 0000 0000 0000 0000 0000
 0140 0000 0000 0000 0000 0000 0000 0000
 0144 0000 0000 0000 0000 0000 0000 0000
 0148 0000 0000 0000 0000 0000 0000 0000
 0152 0000 0000 0000 0000 0000 0000 0000
 0156 0000 0000 0000 0000 0000 0000 0000
 0160 0000 0000 0000 0000 0000 0000 0000
 0164 0000 0000 0000 0000 0000 0000 0000
 0168 0000 0000 0000 0000 0000 0000 0000
 0172 0000 0000 0000 0000 0000 0000 0000
 0176 0000 0000 0000 0000 0000 0000 0000
 0180 0000 0000 0000 0000 0000 0000 0000
 0184 0000 0000 0000 0000 0000 0000 0000
 0188 0000 0000 0000 0000 0000 0000 0000
 0192 0000 0000 0000 0000 0000 0000 0000
 0196 0000 0000 0000 0000 0000 0000 0000
 0200 0000 0000 0000 0000 0000 0000 0000
 0204 0000 0000 0000 0000 0000 0000 0000
 0208 0000 0000 0000 0000 0000 0000 0000
 0212 0000 0000 0000 0000 0000 0000 0000
 0216 0000 0000 0000 0000 0000 0000 0000
 0220 0000 0000 0000 0000 0000 0000 0000
 0224 0000 0000 0000 0000 0000 0000 0000
 0228 0000 0000 0000 0000 0000 0000 0000
 0232 0000 0000 0000 0000 0000 0000 0000
 0236 0000 0000 0000 0000 0000 0000 0000
 0240 0000 0000 0000 0000 0000 0000 0000
 0244 0000 0000 0000 0000 0000 0000 0000
 0248 0000 0000 0000 0000 0000 0000 0000
 0252 0000 0000 0000 0000 0000 0000 0000
 0256 0000 0000 0000 0000 0000 0000 0000
 0260 0000 0000 0000 0000 0000 0000 0000
 0264 0000 0000 0000 0000 0000 0000 0000
 0268 0000 0000 0000 0000 0000 0000 0000
 0272 0000 0000 0000 0000 0000 0000 0000
 0276 0000 0000 0000 0000 0000 0000 0000
 0280 0000 0000 0000 0000 0000 0000 0000
 0284 0000 0000 0000 0000 0000 0000 0000
 0288 0000 0000 0000 0000 0000 0000 0000
 0292 0000 0000 0000 0000 0000 0000 0000
 0296 0000 0000 0000 0000 0000 0000 0000
 0300 0000 0000 0000 0000 0000 0000 0000
 0304 0000 0000 0000 0000 0000 0000 0000
 0308 0000 0000 0000 0000 0000 0000 0000
 0312 0000 0000 0000 0000 0000 0000 0000
 0316 0000 0000 0000 0000 0000 0000 0000
 0320 0000 0000 0000 0000 0000 0000 0000
 0324 0000 0000 0000 0000 0000 0000 0000
 0328 0000 0000 0000 0000 0000 0000 0000
 0332 0000 0000 0000 0000 0000 0000 0000
 0336 0000 0000 0000 0000 0000 0000 0000
 0340 0000 0000 0000 0000 0000 0000 0000
 0344 0000 0000 0000 0000 0000 0000 0000
 0348 0000 0000 0000 0000 0000 0000 0000
 0352 0000 0000 0000 0000 0000 0000 0000
 0356 0000 0000 0000 0000 0000 0000 0000
 0360 0000 0000 0000 0000 0000 0000 0000
 0364 0000 0000 0000 0000 0000 0000 0000
 0368 0000 0000 0000 0000 0000 0000 0000
 0372 0000 0000 0000 0000 0000 0000 0000
 0376 0000 0000 0000 0000 0000 0000 0000
 0380 0000 0000 0000 0000 0000 0000 0000
 0384 0000 0000 0000 0000 0000 0000 0000
 0388 0000 0000 0000 0000 0000 0000 0000
 0392 0000 0000 0000 0000 0000 0000 0000
 0396 0000 0000 0000 0000 0000 0000 0000
 0400 0000 0000 0000 0000 0000 0000 0000
 0404 0000 0000 0000 0000 0000 0000 0000
 0408 0000 0000 0000 0000 0000 0000 0000
 0412 0000 0000 0000 0000 0000 0000 0000
 0416 0000 0000 0000 0000 0000 0000 0000
 0420 0000 0000 0000 0000 0000 0000 0000
 0424 0000 0000 0000 0000 0000 0000 0000
 0428 0000 0000 0000 0000 0000 0000 0000
 0432 0000 0000 0000 0000 0000 0000 0000
 0436 0000 0000 0000 0000 0000 0000 0000
 0440 0000 0000 0000 0000 0000 0000 0000
 0444 0000 0000 0000 0000 0000 0000 0000
 0448 0000 0000 0000 0000 0000 0000 0000
 0452 0000 0000 0000 0000 0000 0000 0000
 0456 0000 0000 0000 0000 0000 0000 0000
 0460 0000 0000 0000 0000 0000 0000 0000
 0464 0000 0000 0000 0000 0000 0000 0000
 0468 0000 0000 0000 0000 0000 0000 0000
 0472 0000 0000 0000 0000 0000 0000 0000
 0476 0000 0000 0000 0000 0000 0000 0000
 0480 0000 0000 0000 0000 0000 0000 0000
 0484 0000 0000 0000 0000 0000 0000 0000
 0488 0000 0000 0000 0000 0000 0000 0000
 0492 0000 0000 0000 0000 0000 0000 0000
 0496 0000 0000 0000 0000 0000 0000 0000
 0500 0000 0000 0000 0000 0000 0000 0000
 0504 0000 0000 0000 0000 0000 0000 0000
 0508 0000 0000 0000 0000 0000 0000 0000
 0512 0000 0000 0000 0000 0000 0000 0000
 0516 0000 0000 0000 0000 0000 0000 0000
 0520 0000 0000 0000 0000 0000 0000 0000
 0524 0000 0000 0000 0000 0000 0000 0000
 0528 0000 0000 0000 0000 0000 0000 0000
 0532 0000 0000 0000 0000 0000 0000 0000
 0536 0000 0000 0000 0000 0000 0000 0000
 0540 0000 0000 0000 0000 0000 0000 0000
 0544 0000 0000 0000 0000 0000 0000 0000
 0548 0000 0000 0000 0000 0000 0000 0000
 0552 0000 0000 0000 0000 0000 0000 0000
 0556 0000 0000 0000 0000 0000 0000 0000
 0560 0000 0000 0000 0000 0000 0000 0000
 0564 0000 0000 0000 0000 0000 0000 0000
 0568 0000 0000 0000 0000 0000 0000 0000
 0572 0000 0000 0000 0000 0000 0000 0000
 0576 0000 0000 0000 0000 0000 0000 0000
 0580 0000 0000 0000 0000 0000 0000 0000
 0584 0000 0000 0000 0000 0000 0000 0000
 0588 0000 0000 0000 0000 0000 0000 0000
 0592 0000 0000 0000 0000 0000 0000 0000
 0596 0000 0000 0000 0000 0000 0000 0000
 0600 0000 0000 0000 0000 0000 0000 0000
 0604 0000 0000 0000 0000 0000 0000 0000
 0608 0000 0000 0000 0000 0000 0000 0000
 0612 0000 0000 0000 0000 0000 0000 0000
 0616 0000 0000 0000 0000 0000 0000 0000
 0620 0000 0000 0000 0000 0000 0000 0000
 0624 0000 0000 0000 0000 0000 0000 0000
 0628 0000 0000 0000 0000 0000 0000 0000
 0632 0000 0000 0000 0000 0000 0000 0000
 0636 0000 0000 0000 0000 0000 0000 0000
 0640 0000 0000 0000 0000 0000 0000 0000
 0644 0000 0000 0000 0000 0000 0000 0000
 0648 0000 0000 0000 0000 0000 0000 0000
 0652 0000 0000 0000 0000 0000 0000 0000
 0656 0000 0000 0000 0000 0000 0000 0000
 0660 0000 0000 0000 0000 0000 0000 0000
 0664 0000 0000 0000 0000 0000 0000 0000
 0668 0000 0000 0000 0000 0000 0000 0000
 0672 0000 0000 0000 0000 0000 0000 0000
 0676 0000 0000 0000 0000 0000 0000 0000
 0680 0000 0000 0000 0000 0000 0000 0000
 0684 0000 0000 0000 0000 0000 0000 0000
 0688 0000 0000 0000 0000 0000 0000 0000
 0692 0000 0000 0000 0000 0000 0000 0000
 0696 0000 0000 0000 0000 0000 0000 0000
 0700 0000 0000 0000 0000 0000 0000 0000
 0704 0000 0000 0000 0000 0000 0000 0000
 0708 0000 0000 0000 0000 0000 0000 0000
 0712 0000 0000 0000 0000 0000 0000 0000
 0716 0000 0000 0000 0000 0000 0000 0000
 0720 0000 0000 0000 0000 0000 0000 0000
 0724 0000 0000 0000 0000 0000 0000 0000
 0728 0000 0000 0000 0000 0000 0000 0000
 0732 0000 0000 0000 0000 0000 0000 0000
 0736 0000 0000 0000 0000 0000 0000 0000
 0740 0000 0000 0000 0000 0000 0000 0000
 0744 0000 0000 0000 0000 0000 0000 0000
 0748 0000 0000 0000 0000 0000 0000 0000
 0752 0000 0000 0000 0000 0000 0000 0000
 0756 0000 0000 0000 0000 0000 0000 0000
 0760 0000 0000 0000 0000 0000 0000 0000
 0764 0000 0000 0000 0000 0000 0000 0000
 0768 0000 0000 0000 0000 0000 0000 0000
 0772 0000 0000 0000 0000 0000 0000 0000
 0776 0000 0000 0000 0000 0000 0000 0000
 0780 0000 0000 0000 0000 0000 0000 0000
 0784 0000 0000 0000 0000 0000 0000 0000
 0788 0000 0000 0000 0000 0000 0000 0000
 0792 0000 0000 0000 0000 0000 0000 0000
 0796 0000 0000 0000 0000 0000 0000 0000
 0800 0000 0000 0000 0000 0000 0000 0000
 0804 0000 0000 0000 0000 0000 0000 0000
 0808 0000 0000 0000 0000 0000 0000 0000
 0812 0000 0000 0000 0000 0000 0000 0000
 0816 0000 0000 0000 0000 0000 0000 0000
 0820 0000 0000 0000 0000 0000 0000 0000
 0824 0000 0000 0000 0000 0000 0000 0000
 0828 0000 0000 0000 0000 0000 0000 0000
 0832 0000 0000 0000 0000 0000 0000 0000
 0836 0000 0000 0000 0000 0000 0000 0000
 0840 0000 0000 0000 0000 0000 0000 0000
 0844 0000 0000 0000 0000 0000 0000 0000
 0848 0000 0000 0000 0000 0000 0000 0000
 0852 0000 0000 0000 0000 0000 0000 0000
 0856 0000 0000 0000 0000 0000 0000 0000
 0860 0000 0000 0000 0000 0000 0000 0000
 0864 0000 0000 0000 0000 0000 0000 0000
 0868 0000 0000 0000 0000 0000 0000 0000
 0872 0000 0000 0000 0000 0000 0000 0000
 0876 0000 0000 0000 0000 0000 0000 0000
 0880 0000 0000 0000 0000 0000 0000 0000
 08

[illegible]

```

0070 CLS:PRINT:ENTER ST-RT ADDRESS
0080 LINE INPUT LHT:AD$
0090 AD=VAL("ENHANCE")
0100 GOTO 1000
0110 PRINT
0120 C=C+1:GOTO LHT#""+LHT#""
0130 PRINT:LINE G=PAGE:AD=LHT#""
0140 PRINT
0150 INPUT ENTER:CHECKSUM=0
0160 PRINT
0170 PRINTG$: "DATA":
0180 FORK=1TOB
0190 ADDRESS=2*(K-1)*B+C OR A*B*P% AND (A*B*Q% OR A*D*Q%) THEN230
0200 DTHGTC=A$:PRINTG$:DTH;
0210 NEXT K
0220 FORK=1TOCOSTERS
0230 CHECK+=VAL("ENHANCE")(DTH,K,D) :AD=AD+1
0240 NEXT K
0250 IF K=0 THEN CLS:PRINT:"DATA INCORRECT!! RE-ENTER":GOTO150
0260 FORK=1TOCOSTERS
0270 PAGE=D:VAL("ENHANCE")(DTH,K,D) :AD=AD+1
0280 NEXT K
0290 IF AD#HOC01 THEN CLS:PRINT:"FINISHED":END
0300 GOTO100

```


Sliding Graphics

Pam D'Arcy slips into something a little more BASIC.

PERHAPS it is a sign of the maturity of the Dragon and its users that these days "Dragon User" features an abundance of machine code articles and, excellent though they may be, programs containing really few dunks. However, I still come across many readers who are not interested in machine code and others who are put off by long listings, so once again I am offering a non-anale game of minimum length, written entirely in BASIC, capable, of your own enhancements to suit. DEM lines may be omitted for even faster completion. For those who like rather more than just a listing, some explanation of the Dragon BASIC graphics statements used completes the article.

Slide Puzzle Program (Listing 2)

The program takes the existing contents of graphics pages 1-4, copies them to pages 5-8 where it treats them as a 4x4 grid as in a sliding puzzle, scrambles the 'tiles', then waits for the player to reconstitute the original picture by moving the 'tiles' via the 'blank' square using the arrow keys. Movement is as per the plastic puzzle, that is, a 'tile' is moved into the 'blank' space. There is an optional option (x-y) to allow the totally lost to start again!

Any existing PAGED screen may be used, although 'tile edging' and a contrasting coloured 'blank' square may need to be incorporated for easier operation. Instructions for doing this are detailed below. The 'default' position of the blank square is the top left of the screen. Meanwhile, a quick picture may be obtained from Listing 1.

A Picture Of A House (Listing 1)

I recommend that Line 540 is typed in first then to RUN the program after typing in each LINE/CIRCLE/PAGE statement which will show you the effect of each graphics statement and will confirm the validity of the statement as typed rather than tedious debugging after typing in the whole. When completed, delete line 540 if you intend to append Listing 2 to form a single program.

An Existing Picture

Any existing PAGED graphics picture currently in graphics pages 1-4 may be used. You may have one available from a graphics generator program, light pen or touch pad creations, from interrupting (BREAK-RESET) a game containing a nice graphics display or one designed and drawn up by yourself or your pen-mating.

If you do not have a copy of the screen stored on tape or disk, I recommend that as the first step so that it can be reloaded another time, including if it gets messed up following these instructions! If needed,

Table 1 offers assistance with saving and loading the graphics screen.

Listing 2 (comprising an edited LINE 50, new LINE 58 and existing LINE 5 440-540 of Listing 1) is a program that will show in the tile-edging and 'blank' square in the top left position of the screen. To change the blank's colour, the first parameter of the COLOR statements should be changed to the number of the required colour. To change the position of the blank square, the appropriate co-ordinates as given in Table 2 should be used in LINE 508. The default values for the position of the blank square in the Slide Program will need to be amended accordingly (see below).

LINE 50 makes a quick 'safety' copy of graphics pages 1-4 to pages 5-8 when RUN. Should you want to change something after the first RUN, a re-load of the original screen from tape or disk is unnecessary if you also edit LINE 58 to ... POOPY N=4 TO N ... to restore the original copy when the program is subsequently RUN.

Having set up the tile edging and blank square, save the masterpiece to tape or disk for future loading to use with the Slide Program.

Program Techniques

Various 'default' values can quickly be changed to suit your requirements. These are coded on LINE 600:

CS=colour set (when RUN, key C changes colour set anyway)
 DX, DY=default blank square co-ordinates of the 4x4 grid, 0-3 across (DX) by 0-3 down (DY).

thus 0.0=top-left; 3.0=top right; 0.0=bottom-left; 3.0=bottom right etc. M1=value that is subject of RAND to determine number of the movements (when added to 8) to jumble the picture.

thus the initial default jumble is between 7 and 16 moves. (I use M1=16 for my children).

The GET statement copies a described rectangle from the screen display into an area of memory called an ARRAY 'Sizable'. That Array Variable can then be copied into different part of the screen using the PUT statement. In the Slide Program, we need to swap the blank square 'rectangle' with an adjacent 'rectangle' of a 'picture tile'. The contents of the blank square 'rectangle' never change so LINE 740 copies it to the array 'B5' where it stays for the duration of the run.

The important thing about the size of the array variable for GET PUT is that the manual is wrong and greatly overstates the required size.

The width of the 'tiles' in pixel points for each of the four 'tiles' across the screen is 256/4=64 (variable PX). The depth of the 'tiles' in pixel points for each of the four 'tiles' down the screen is 192/4=48 (variable PY). According to the manual, this would require an array DIM B5(63,47) to store a copy of any of the 'tiles'. In fact, the graphics data is tightly packed into the array and a formula for calculating the required size that will work for all PAGED and GET PUT options is:

TABLE 1
SAVE/LOAD FROM/TO GRAPHICS PAGES 1-4

System	SAVE	LOAD
Cassette only	CURVEM "SCREEN", 1536, 1536+1544-1, 1544	CLOADM "SCREEN"
DragonDOS 1.0	SAVE "SCREEN", 5072, 5072+5144-1, 5144	LOAD "SCREEN SAV"
DragonDOS 4.0/5 & Commodore 2.0	SAVE "SCREEN", 5072, 5072+5144-1, 5144	LOAD "SCREEN SAV"
DemoDOS	SAVEM "SCREEN", 1536, 1536+1544-1	LOADM "SCREEN"

TABLE 2
X, Y CO-ORDINATES TO FILL 'RECTANGLE' WITHIN 'EDGED TILE'

X,Y	0,0	0,1	0,2	0,3	1,0	1,1	1,2	1,3	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3
0,0	0,0	0,1	0,2	0,3	1,0	1,1	1,2	1,3	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3
0,1	0,1	0,2	0,3	0,4	1,1	1,2	1,3	1,4	2,1	2,2	2,3	2,4	3,1	3,2	3,3	3,4
0,2	0,2	0,3	0,4	0,5	1,2	1,3	1,4	1,5	2,2	2,3	2,4	2,5	3,2	3,3	3,4	3,5
0,3	0,3	0,4	0,5	0,6	1,3	1,4	1,5	1,6	2,3	2,4	2,5	2,6	3,3	3,4	3,5	3,6
1,0	1,0	1,1	1,2	1,3	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3	4,0	4,1	4,2	4,3
1,1	1,1	1,2	1,3	1,4	2,1	2,2	2,3	2,4	3,1	3,2	3,3	3,4	4,1	4,2	4,3	4,4
1,2	1,2	1,3	1,4	1,5	2,2	2,3	2,4	2,5	3,2	3,3	3,4	3,5	4,2	4,3	4,4	4,5
1,3	1,3	1,4	1,5	1,6	2,3	2,4	2,5	2,6	3,3	3,4	3,5	3,6	4,3	4,4	4,5	4,6
2,0	2,0	2,1	2,2	2,3	3,0	3,1	3,2	3,3	4,0	4,1	4,2	4,3	5,0	5,1	5,2	5,3
2,1	2,1	2,2	2,3	2,4	3,1	3,2	3,3	3,4	4,1	4,2	4,3	4,4	5,1	5,2	5,3	5,4
2,2	2,2	2,3	2,4	2,5	3,2	3,3	3,4	3,5	4,2	4,3	4,4	4,5	5,2	5,3	5,4	5,5
2,3	2,3	2,4	2,5	2,6	3,3	3,4	3,5	3,6	4,3	4,4	4,5	4,6	5,3	5,4	5,5	5,6
3,0	3,0	3,1	3,2	3,3	4,0	4,1	4,2	4,3	5,0	5,1	5,2	5,3	6,0	6,1	6,2	6,3
3,1	3,1	3,2	3,3	3,4	4,1	4,2	4,3	4,4	5,1	5,2	5,3	5,4	6,1	6,2	6,3	6,4
3,2	3,2	3,3	3,4	3,5	4,2	4,3	4,4	4,5	5,2	5,3	5,4	5,5	6,2	6,3	6,4	6,5
3,3	3,3	3,4	3,5	3,6	4,3	4,4	4,5	4,6	5,3	5,4	5,5	5,6	6,3	6,4	6,5	6,6

PS=Width of rectangle involved in
PIXEL POINTS

PT=depth of rectangle involved in
PIXEL POINTS

Then ARRAY VARIABLE size=INT (PWT
(PWT*PT)+7)/8+4/8)

Thus, typing in the Dragon the above
information in COMMAND MODE

PS=64 <ENTER>

PT=48 <ENTER>

PRINT INT(INT((PWT*PT)+7)/8+4/8)
<ENTER>

reveals a required Ddd Array size of 77 (as
even the B&B&D slightly exceeds the
minimum requirement).

There is no harm in overestimating the
required size, apart from wasting memory,
but there is every harm in understating the
size. Also, unless the normal use of arrays in
BASIC where if an array that has not been
specifically DIMensioned is encountered,
BASIC automatically allocates a default
size of 16, the array MUST be defined in a
DIM statement as BASIC will flag an
ERROR and NOT default define the
ARRAY in these circumstances. The pur-
pose of the LD variable checked in line 820
is to avoid the start line move when jumping
the picture to simply mirror the previous
move, thus negating some of the effect of
the moves.

The GETPUT subroutine in lines 1040-
1080 swaps the tile to be moved with the
adjacent 'blank' square. At this point, the
4*4 grid co-ordinates are held in MX and
MY for the 'picture tile' being moved and BX
and BY for the current position of the 'blank
square'. First, the 'picture tile rectangle' is
GOT into the MY Array variable. The 'blank
square rectangle' copied into array variable
BS at the start of the routine PUT over that tile
on the screen. The 'picture tile rectangle' in
array MS is then PUT over the 'blank

square rectangle' but it is being moved to
the blank square co-ordinates as they
updated to its new position (line 1070).

Avoiding FC Errors

If the Side Program is abandoned by, say,
pressing BREAK rather than using the 'H'
option, and the next PCLEAR statement
used is other than PCLEAR0, error directly
from the keyboard or even in a newly
loaded program prior to a PMODE state-
ment being issued, an FC ERROR will
occur.

This is because the Side Program was
currently set to PMODE 3.5 thus using
graphics pages 5-8 and the BASIC inter-
preter safeguards the graphics pages
needed by the current PMODE setting. A
lower PCLEAR figure will not be allowed
until the machine has had its PMODE
setting suitably reset. Thus when the
program is quit through its in-built option
(*), the PMODE setting is set to the lowest
option requiring just one graphics page,
PMODE0.5 prior to the END statement line
990. The PCLEAR0 is reset to PCLEAR4
to release the unneeded 'working' pages of
the side puzzle, but retaining the 'master'
pages 5-8 intact in case the program is
re-RUN. A BASIC program is physically
moved down into the freed graphics pages
as soon as a lower PCLEAR statement is
issued; by typing the program using the
'H' option, typing in PCLEAR1<ENTER>
from the keyboard then RUN<ENTER>.
The visible graphics screen corruption is
where the Side Program was moved to
following PCLEAR0. It is physically copied
to higher in memory when a greater number
of graphics pages than at present are
reserved with PCLEAR. It is quite a good
idea to commence graphical programs with
a PMODE.1 so that FC Errors are avoided

when setting up graphics requirements
regardless of the state that a previous
program may have left the machine in.

Picture Programming

I have space rather longer than allotted on
the Side program so will just mention one
or two final points regarding the Picture
Program. In PMODE3, the size of the
colour unit is 2 pixel points wide by 1 pixel
point deep. The co-ordinates used are still
based on the highest resolution graphics
screen, being 256 pixel points across (and
192 points down). They are addressed as
0-255 and 0-191 respectively. If an odd
pixel co-ordinate is specified when refer-
encing a point across the screen (= X
co-ordinate), that co-ordinate has 1 sub-
tracted from it when accessing the screen.
That is, as in the program where on X
co-ordinate such as 255 is used (line 681),
the block of colour pointed in comprises the
next pixel points 254 and 255. The size of
the colour unit is why a STEP of 2 is used in
line 680.

LINE statements with no concluding B or
BF parameter result in a line being drawn
from the first pair of X, Y co-ordinates to the
second pair. LINE statements concluding
with a B alone means 'draw a rectangle
(Box) in outline only', the pairs of X, Y
coordinates defining the diagonally oppo-
site corners of the rectangle. The BF
parameter on the LINE statements means
'draw and Fill this rectangle (Box) with the
current foreground colour'. Only rectangles
can be automatically filled with colour.
Other shapes, such as the circle of sun-
shine and the roof of the house, need to be
PAINTed after drawing the outline. The
areas to be PAINTed needs to be completely
outlined with a defined single colour (border
otherwise the 'paint' will spread alarmingly)

Listing 1

```
10 REM ***** PARTIAL OF A HOUSE FOR SIDE PUZZLE PROGRAM
20 PICTURE=0:COL=0:ROW=0:CONTINUE=0
30 REM ***** DRAW
40 PUT 0
50 REM ***** GET
60 GOTO 80 IF CONT=0:GOTO 900 IF PUT
70 REM MOVE:CON=CON+1:GOTO 1000 IF CON=0:GOTO 800
80 REM:CON=0:GOTO 800 IF CON=0
90 REM:CON=0:GOTO 800 IF CON=0
100 REM:CON=0:GOTO 800 IF CON=0
110 REM:CON=0:GOTO 800 IF CON=0
120 REM:CON=0:GOTO 800 IF CON=0
130 REM:CON=0:GOTO 800 IF CON=0
140 REM:CON=0:GOTO 800 IF CON=0
150 REM:CON=0:GOTO 800 IF CON=0
160 REM:CON=0:GOTO 800 IF CON=0
170 REM:CON=0:GOTO 800 IF CON=0
180 REM:CON=0:GOTO 800 IF CON=0
190 REM:CON=0:GOTO 800 IF CON=0
200 REM:CON=0:GOTO 800 IF CON=0
210 REM:CON=0:GOTO 800 IF CON=0
220 REM:CON=0:GOTO 800 IF CON=0
230 REM:CON=0:GOTO 800 IF CON=0
240 REM:CON=0:GOTO 800 IF CON=0
250 REM:CON=0:GOTO 800 IF CON=0
260 REM:CON=0:GOTO 800 IF CON=0
270 REM:CON=0:GOTO 800 IF CON=0
280 REM:CON=0:GOTO 800 IF CON=0
290 REM:CON=0:GOTO 800 IF CON=0
300 REM:CON=0:GOTO 800 IF CON=0
310 REM:CON=0:GOTO 800 IF CON=0
320 REM:CON=0:GOTO 800 IF CON=0
330 REM:CON=0:GOTO 800 IF CON=0
340 REM:CON=0:GOTO 800 IF CON=0
350 REM:CON=0:GOTO 800 IF CON=0
360 REM:CON=0:GOTO 800 IF CON=0
370 REM:CON=0:GOTO 800 IF CON=0
380 REM:CON=0:GOTO 800 IF CON=0
390 REM:CON=0:GOTO 800 IF CON=0
400 REM:CON=0:GOTO 800 IF CON=0
410 REM:CON=0:GOTO 800 IF CON=0
420 REM:CON=0:GOTO 800 IF CON=0
430 REM:CON=0:GOTO 800 IF CON=0
440 REM:CON=0:GOTO 800 IF CON=0
450 REM:CON=0:GOTO 800 IF CON=0
460 REM:CON=0:GOTO 800 IF CON=0
470 REM:CON=0:GOTO 800 IF CON=0
480 REM:CON=0:GOTO 800 IF CON=0
490 REM:CON=0:GOTO 800 IF CON=0
500 REM:CON=0:GOTO 800 IF CON=0
510 REM:CON=0:GOTO 800 IF CON=0
520 REM:CON=0:GOTO 800 IF CON=0
530 REM:CON=0:GOTO 800 IF CON=0
540 REM:CON=0:GOTO 800 IF CON=0
550 REM:CON=0:GOTO 800 IF CON=0
560 REM:CON=0:GOTO 800 IF CON=0
570 REM:CON=0:GOTO 800 IF CON=0
580 REM:CON=0:GOTO 800 IF CON=0
590 REM:CON=0:GOTO 800 IF CON=0
600 REM:CON=0:GOTO 800 IF CON=0
610 REM:CON=0:GOTO 800 IF CON=0
620 REM:CON=0:GOTO 800 IF CON=0
630 REM:CON=0:GOTO 800 IF CON=0
640 REM:CON=0:GOTO 800 IF CON=0
650 REM:CON=0:GOTO 800 IF CON=0
660 REM:CON=0:GOTO 800 IF CON=0
670 REM:CON=0:GOTO 800 IF CON=0
680 REM:CON=0:GOTO 800 IF CON=0
690 REM:CON=0:GOTO 800 IF CON=0
700 REM:CON=0:GOTO 800 IF CON=0
710 REM:CON=0:GOTO 800 IF CON=0
720 REM:CON=0:GOTO 800 IF CON=0
730 REM:CON=0:GOTO 800 IF CON=0
740 REM:CON=0:GOTO 800 IF CON=0
750 REM:CON=0:GOTO 800 IF CON=0
760 REM:CON=0:GOTO 800 IF CON=0
770 REM:CON=0:GOTO 800 IF CON=0
780 REM:CON=0:GOTO 800 IF CON=0
790 REM:CON=0:GOTO 800 IF CON=0
800 REM:CON=0:GOTO 800 IF CON=0
810 REM:CON=0:GOTO 800 IF CON=0
820 REM:CON=0:GOTO 800 IF CON=0
830 REM:CON=0:GOTO 800 IF CON=0
840 REM:CON=0:GOTO 800 IF CON=0
850 REM:CON=0:GOTO 800 IF CON=0
860 REM:CON=0:GOTO 800 IF CON=0
870 REM:CON=0:GOTO 800 IF CON=0
880 REM:CON=0:GOTO 800 IF CON=0
890 REM:CON=0:GOTO 800 IF CON=0
900 REM:CON=0:GOTO 800 IF CON=0
910 REM:CON=0:GOTO 800 IF CON=0
920 REM:CON=0:GOTO 800 IF CON=0
930 REM:CON=0:GOTO 800 IF CON=0
940 REM:CON=0:GOTO 800 IF CON=0
950 REM:CON=0:GOTO 800 IF CON=0
960 REM:CON=0:GOTO 800 IF CON=0
970 REM:CON=0:GOTO 800 IF CON=0
980 REM:CON=0:GOTO 800 IF CON=0
990 REM:CON=0:GOTO 800 IF CON=0
```


Expert's Arcade Arena

Write to 'The Expert' at Dragon User
10-13 Late Newprint St, London WC2E 9RF,
with all your arcade tips and hints.

GOOD DAY to you, all and welcome to the fourth arcade column and once again may I thank you for all the letters you have sent. I have thought a worthwhile to share them (with a few edits) for their safety. However, should you ever wish to see anything you sent, then again please send an SAE, or alternatively leave ten thousand pounds in a hollow tree in Dagenham by Monday or I'll start sending them back. In bits. Heh, heh, heh.

Firstly the winner of a year's subscription is a Mr David Barclay of Dumfries, Scotland (you know, the party at the top with the haggis and the loch) who correctly named pictures (a) and (b) as 3-D Scudlab Attack and Hissie Goes (Swingingly). Congratulations David and tough (conceded) to the rest of you, especially those of you who can't spell. Scudlab.

Now then, moving at the speed of light on to Total Eclipse (which isn't strictly an arcade game but still seems to be the subject of several thousand letters to me) and two different ways of helping yourselves along with this game.

Firstly, you write to Eclipse-Pearl who expressed an interest in all the recent 5000 conversion. It selling board games for a couple of quid at various points in the game. This sounds to me like a pretty groovy idea but it's up to you to nag them!

If you don't feel like doing that then there is another solution. Over to J. Brown of Buckinghamshire with his 'Total Eclipse Savings Editor'. Mr Brown's program will only run on a Dragon 64 and as I only possess a Dragon 32 (unless any company out there really feels benevolent and wants to butter up a writer) I have not tested it.

Mr Brown has sent complex instructions but the program is very easy to operate and they're not really necessary. Here are the prominent excerpts:

"The program only works on a 204 in 84k mode but can be typed in on a 32. Before loading however the 84k mode must have been selected. . . . For use with option 8, locations 32115 to 32122 for investing, 32104 to 32130 will change the number of shares, disks, and pills you are carrying. . . . Within option 8 'Q' returns you to the memory and 'B' will hold the listing or when held down make the listing slower. . . . The maximum number of credits is 4290000799. Any more and the program will crash!"

Thanks you very much, Mr Brown, some more Total Eclipse next month, now to The Dark Pit and the map from Simon Dickson. To be brutally honest I think that there is an error on this map as looking at it there seems to be no way to get to the exit, but my copy is mangled and so until my new one

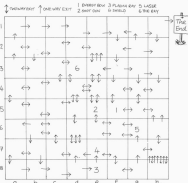
arrives I can't tell it. However, rather than hold it back from you I've decided to go public to help the map and please, don't send me any more maps of this, or of Jet Set Willy as although the thought is very nice, they're no longer necessary! However, do keep sending maps of any game you have (JBooker Kingdom, anyone?), I have even given consideration to publishing maps of adventure games (look out Mike Goirand, this is a takeover bid) but please, please, please, if you want them returned make sure your name and address is on the map as well as the letter you send with it. Better still, keep a photocopy.

Moving on, an appeal now from Jonathan Harrop of Oxon: "In Time Bandit I have successfully completed 40 on 'The Time Gate' and have been returned to 1A, but I failed to do this on any of the other screens so I have not found out what the secret message is. I am now completely bored and thoroughly sick with this game. Can you please tell me what the message is?" Answers on a postcard to the usual address!

And so to a man who is becoming a co-writer on this column! Mr M. R. Vance who reliably informs me that his name is "Mike" ("Mike the Brave" or "My You" . . . doesn't explain the 'H' Model 550, Mike tells us that the access code for Beambuster is

THE DARK PIT

MAPPED
by
SIMON
DICKSON



[illegible]

Regan said that he can't go any further and that holding down **CLEAR** and **M** gives extra lives up to a total of 100. He has asked me to publish his address so that he can enter into communication with other (dare I say the word?) hackers. But I'm not sure if this is such a good idea as then people might write to him instead of me and I'd be all alone in the world and go slowly mad here in my packed cell. However, I will publish his address as soon as he sends me his phone number! This is not, you understand for any seditious reasons, but so that I can clarify some points that have arisen from his letters, although, perhaps we could go out for a quiet drink afterwards, and then back to my perchouse flat for a wild night of loud music, gambling, and, of course, hacking. And we could always have a **computer** for fun! (One for the editing room there Helen!) I love you, my dear, yours, (P)

Sorry, I forgot myself there, some more clues and hints for games, in no particular order, and from a fair Anant Patel of Coventry.

Crazy Painter: On letting the space invader reach the bottom, when your paint reaches the zero level it will also have an

infinite supply. (I can't get this to work, Anant... maybe someone else can.)

Hungry Hound: The guard can't get you when you're both in the tunnel so you can pass straight through him.

Joe Lowe's Cricket: Using a "Quickshot II" joystick, when playing against the computer, position the bowler as far down as you can, then position the bowler's arm in a top-left position and bowl. A large percentage of the time you'll be able to bowl a full toss at the top of the off stumps and get it out.

Finally, also from Ansat, he says if you load games with the cartridge "AUDIO INCREDIBLE AUDIO CD" you can meet a music cassette in the player and rap along to, and I quote, "Talking Heads, Simple Minds or Tom Tom Club." Personally, I find Paul is true best to Tangerine Dream, Robert Wyatt, Tom Waits (Discworld Trombones and Rain Dogs especially) and early David Bowie. Has anyone found the ultimate record to play along with? If so, you know the address.

Now then, is a topic which has been winding me up for some months now: the future in cinema, and the conclusion is:

volved in poking them. Quite simply, and there is not a hint of a joke here, I mean that I am pretty fed up with every game's hero being both white and male and getting the girl at the end of the game. I have never managed to impress women by killing off a race of space invaders. I don't think many women are interested in a cosmic space hero as a partner for life! How about a little of sensitivity, how about a black hero (a heroine)? How about racism and sexism?

Right, that should start the letters flowing. If you agree or disagree, or you know of games which break the convention, write to me, and together we'll lead the Dragons' revolt to revolution.

On all levels is a little coffee shop I know off Leicester Square. (Back to square one ... 2011)

That's about it for this month. Please keep writing, tell me what you'd like to see, stop sending me your high scores, if not prepared to massage your egos, and let's try and get the Dragon game crew thinking about what it really wants to play, remember, software makers read this column too.

Chang, J. and S. G. Li. 2003. *Water Pollution in China*. Beijing: China Water & Power Press.



I HAVE to admit to being all at sea this month, and the blame for that has to go to Microdeal and their new adventure, *Aquanaut 471*, but more of that later. First to reader James Boyden and a query he's got on an adventure called ... wait a minute, *Aquanaut 4717*? Hang on, James, you might at least wait for me to review them before you start getting stuck in them. James wants to know where the seaweed is, I've had a mind not to tell him. (Those who said I've only had a mind anyway, kindly leave the page.) Have you encountered the Mubari yet, James? If you can't get past that bewitching go W-N-E-S (TO REACH THE SEAWEED) — and then you've got to know how to get it. SRETTUO TUALPHTM TI TUC.

Syzygy

James also wants to know how to get to the planet on Syzygy, the answer to which was given last month, but in case you missed it (how dare you!) you enter the co-ordinates then put the laser, the co-ordinates for the planet being 0-4-1-5 (and that's the right way round just for a change). Finally on *Water Factor*, how do you open the safe? You need an S&B to Tom Wilkinson of 13 Shaftesbury Ave, Hat, Humberdale, that's what. No, Tom isn't a salescracker, or at least I don't think he is, but he has just written to me claiming his *Dragon User* merit card for having solved *Miner's Factor*.

Mr. A. M. Marks of 10 Molecombe, Penwyther, Cambourne, Gloucs NP44 4HE wants to swap or sell *Syzygy*, the *Hulk* and *21 Doctors*, if anyone is interested, while Alan (legible) Signature of Redbrix, Cornwall wants to know how to pass the force field on the garden planet. A good job I know he means *Trekboar*, in which protection from the force field is given by TRLJUMA EHT.

A *Trekboar* tip from P. Sheppard of Chesham, for people wondering how to deal with the second spider. The answer: RECPHS DMOCES ON SI EREHT. To make sure of that you must deal with the first spider properly, and take it when dead to MOOR SDORACH XE EHT. This GDSG-TUC NOTTUS EHT ESSEPH. This reader also asks which *Dragon* adventures will also run on the Tandy computer, and I think it would be a good idea if we could compile a list of those for some future column, so

that all Tandy owners out there write it with a list of those *Dragon* adventures you know for sure will run on the Tandy, for the benefit of everyone.

Yet more help needed on *Trekboar*, this time by Adrian Hall of Huddersfield, and how to stop the Kendera flower from dying is the first question. Don't plant it too early in the game, Adrian, as it's almost the last thing you do. Secondly, how to get the ice to water the flower when necessary: TERNALB EHT NI TI YHRAC. In *Caveins of Doom* how to build the raft when you've got the logs, hammer, nails, saw and beams: EROH EHT DEENLUITS UOY. And how to explore underwater without running out of air: PU-N-HW-MANNO DO.

If anyone knows how to get over the oil fire on the third level of *The Plunderer's Adventure*, can they write and tell Stephen Heaton, 4 Bankcroft, Langton, Preston, Lancs PR1 6AL, and Stephen also wants to know how to obtain food from the dinner clock in *Jungleposition*, which is straightforward enough: 'DOOP IM EHT' DORD REHND OT YAS. You don't even need to say please.

Mr B. W. Le Roux writes to me on his flash Amstrad PCW from April Cottage, Cliff End Lane, Pott Leval, Nr Hastings, East Sussex TN35 6GF, and has come up with a potential money-making scheme. He says I should print pages of clues every month but make them coded instead of merely backwards — then I can make a few bob by selling the code-books. Nice thinking, Mr Le Roux, but I wonder if the new editor would let me get away with it? I can give a clue backwards, forwards, cryptic or in binary for this reader's problem in *Syzygy*, as he's blundered into a big forest where the light is a strange colour and he can't do anything. Can anyone shed any un-strange light on that problem? The same reader is also having light trouble in *Caveins of Doom*, wondering how to reach the broken lamp. He says he's tried using the oil to unstick it, the rope to improvise a walk, and has even taken the pickles out of the jar to try to put the lamp in there to shield it from draughts, but all to no effect. I'm not surprised, either. Why? EREHWESLE PMAL MEKORWNU NA SI EREHT. Why hasn't this adventurer discovered yet? It looks from the map as if he hasn't MOOR LLAMS EHT NI LLAW EHT DENNAXE. Mr Le Roux also offers help on *Pellegrina's Diary*, amongst others,

which not many people claim to have finished.

Where is the bomb hidden in *Mings of Mar*, asks A. Court of Birmingham, RAL-LEC EHT NI, page 1, where you go HTRON TS&E HTUCS HTUCS then R&M&TWOC XE with H&M&FW DMA MUBH&M&LA and finally TS&E DO.

All this backwards writing is taking its toll, so I'll give myself a break by looking at *Aquanaut 471*. Under the *Doomed Sea*, another of Microdeal's imported American adventures converted for the *Dragon*, and also available for the Tandy. This takes place in the 21st century, where Jacques Cousteau's vision of underwater cities has become a reality, and you play the part of a high-ranking member of the *Dominic Federation*. It seems France's brother of 1718land, as that's where you're headed when the adventure begins. (Rough you won't know what the trouble is if you're contacted Huey-14, the *Dome's* service chroil, who sent out the SOS.)

You only have a limited number of moves to find the *Dome* when travelling around the various underwater scenarios, but luck led me to it first time — bad luck, I think, as this then brings you to the first of what might be called arcade games, if arcades had ever been so primitive. You have to use your joystick to manoeuvre your submarine across the screen through bubbles floating to the surface to reach the *Dome's* landing spot. It's awkward to do, and rather annoying if, like me, you'd been looking forward to getting stuck into a new adventure. Once you're through, be sure to save your game so you don't have to go through the silly game again.

Underwater city

Here the adventure proper does begin, and you can start to map out the underwater city, wondering what you do with the last pipe and the memory grid that you soon find. Not that you worry for long as, oh no, it's another arcade game ... just where I was starting to get interesting, too. This time you race your jet self up the screen through a maze past a series of moving robots.

I'm afraid I found these arcade elements tedious and disruptive to what would otherwise be a promising adventure with the high standard of moving graphics we've come to expect from the likes of *Trekboar* and

Mini Maths

Gordon Lee sets the puzzles — but who will find the totals?

FROM TIME to time on this page we present a series of unrelated puzzles that the reader might be interested in tackling, purely for his (or her) own enjoyment. Here are five such puzzles, but the solver should proceed with caution as not all of them are necessarily solvable by the use of the computer.

1. Using each of the nine digits 1 to 9, once and once only, arrange them to form a perfect square. For example, one such arrangement might be 382945761, the square of 19569. There are many other arrangements of the nine digits possible, but can you find a) the lowest possible number, and b) the highest?

2. As in question 1, use the nine digits 1 to 9, but this time arrange them to form two nine-digit prime numbers: a) the lowest possible, and b) the highest.

3. A farmer has a circular field with a diameter of 300 feet. The field is enclosed by a fence. Inside this field he set to the perimeter fence is a goat on a length of rope. If the rope is long enough to allow the goat to graze exactly half the area of the circular field, how long is the piece of rope? (You should assume that the goat can reach exactly to the end of its tether).

4. The number 4 is very interesting. It is a perfect square, and the two integers each side of it, 3 and 5, are both prime. Can you find the next highest square number that has this property?

5. Consider the following alphametic:

DRAGON
USER

In alphametics, each letter represents a certain digit, whatever it occurs. Similarly, unlike letters represent unlike digits. So in the puzzle given we have a six-digit number with all digits different (as represented by the word 'DRAGON'), divided by a four-digit number, also with different digits ('USER'). Note however that the second digit of the numerator is the same as the final digit of the denominator, as represented by the letter 'R'. The result of this division is represented by the two asterisks. Of this number nothing is known except that it consists of two digits, which may, or may not, be alike. However, if this value is cubed, and the digits of this cube exchanged for letters using the same substitution as in 'Dragon User', the result will be a familiar English word. Can you find the correct values of the letters and the word is produced?

For the competition this month, we require just the solution to this final question (No. 5).

The solution to question 1 follows, as if you wish to tackle it yourself, read no further. The solutions to questions 2 to 4 will be published next month.

1. The lowest and highest numbers are 138542769 and 932187654, the squares of

11826 and 30534 respectively. One method of solution is by running the program listed below. Here, the values in the range 11112 to 31426 are squared and the resulting figure is tested to determine if it is a compound of the nine digits, 1 to 9. In order to do this, the numerical value needs to be converted into a string variable so that individual digits can be selected for examination. Note the second command in line 30: `55=mid$(55,2)` which removes the first character of the string. On the 'Dragon', as on some other machines, when a numeric variable is converted to its string equivalent, an extra 'invisible' character appears in the first position in the string. This character holds the imaginary plus sign in front of all positive numbers. (If the number were negative, the minus sign would appear here, as we would expect). Consequently, we need to remove this extra character from the string, before examining its contents.

Having done this, the remaining nine digits need to be tested to determine that there is no zero present, and that no digit is duplicated. The test for duplication is not difficult as every character in the string can

be compared with every other, but it is time-consuming to test every one of the possible squares in this way. Fortunately there is a short cut available. In any arrangement of the digits 1 to 9, the sum of the digits must equal 45, and the product must equal 362880. Consequently, it is simply necessary to move along the string taking each digit in turn and finding the respective sum and product. Any value not in agreement with the expected values can be rejected. Note that this procedure will not guarantee that every number which passes the test does contain the nine digits, but it will reject all those which do not have sums of 45 and products of 362880. If the listed program is run it will print out all 30 squares which are made up of the digits 1 to 9, but it also includes a further two containing duplicated digits. However, these are easily spotted and can be eliminated. Note that the question, as stated, only asked for the lowest and highest values, so if it is suit until the first answer is printed (the lowest value), and then line 10 is amended to: `10 FOR N=31426 TO 11112 STEP -1`, the program can be rerun to compute the highest value.

Prize

Guess what we've got this month? A real tuck-in for the bookworms. For those of you who welcomed your chance of getting a huge free copy of *Total Eclipse* sent (spaced of light) to one against *Eclipse-Fest* are looking their complete confidence in the



1.3 version by sending us here 20 copies for our August prizewinners. Our man on the spot (listed above) *Total Eclipse* ... reads of *Traveller* saved in frustration when the Universe looked up a thousand light years from home. Now it seems that the Universe is rolling again.

Rules

To solve this puzzle and win a prize, you must give the answer and show how you arrive at it, using a Babbage program. Don't send cassettes, just a printout, and any explanation you want to offer (and our kids to support won't be accepted). Please mark your envelope **AUGUST COMPETI-**

TION, and don't forget to put your name and address on your entry and send it to reach us by mid-September at the latest.

This month's prizewinner, complete the following phrase: "You'll never score in the Universe: ..." It can be a lineolite if you mean — just so long as it begins "You're never alone in the Universe: ..."

May Winners

Our trusty trophy drawer this month contains 20 copies of *Elo's* hit *Kung-Fu* — The Master, and these are on their way to J. B. Slinger of High Wycombe, Andy Beale of Wallington, Phil Sapota of Liverpool, E. C. Harriest of Ebb, D. C. Lee (not the Geo C. Lee) of Barnsley, J. L. Clark of Portsmouth, Ralph E. Newman at Penridge, S. P. Hoot of Royston, P. D. Macdonald of Taplow, Simon Aubrey of Beardon, D. A. Hunt of Canterbury, G. R. Barber of Sutton Coldfield, E. A. Newman of Addlestone, M. W. Stanton of Tewkesbury, Mark Heaps of Wallington, P. A. Bennington of Stood, P. Fairclough of Wincoburn, M. C. Rogers at Feltham, E. K. Jones of Cardiff and D. C. Paulsen of Potsgrove.

Solution

Many calculated correctly that 5 to the power of 36 is 100144276840626546171649568... but not everyone remembered to say that the odds involved are either 1 in ... 055 or ... 055 to 1. Prizewinners 'drew' heavily on *Unearthed Dragons*, but the Editor's favourites 1 get a look out of my *Dragon*, because it's perfect for 'whatever ju da'. Quah!

