STOP PRESS

DRAGON

Voice of the Dragon

The Dragon 32 home computer was launched in August 1982. Since that date it has become the best selling home computer in its price bracket. There are now over 40,000 Dragon users in the United Kingdom and by the end of 1983 this figure is expected to exceed 150,000.

This represents an enormous number of Dragon enthusiasts around the country who are enjoying the power and versatility of the Dragon 32.

To help keep Dragon users informed of the very latest developments in Dragon technology, including the introduction of new peripherals, the latest on software releases and even the development of new Dragon computers, an official newsletter will be sent directly to Dragon users quarterly in 1983.

Aimed primarily at existing Dragon 32 owners, the newsletter will be used to create a forum for articles on the many facets of programming the Dragon.

If you would like to contribute programs, comments, suggestions, hints or stories on using the Dragon 32 then please write enclosing details to:

The Editor, The Newsletter, c/o Dragon Data Ltd., Kenfig Industrial Estate, Margam, Port Talbot, West Glamorgan.

Submission of Programs

All program suggestions should enclose a program listing and a letter explaining its basic structure. Long programs should be submitted on a cassette tape which is itself clearly labelled with your name and address, together with a self addressed envelope for its return. List program lines with a maximum of 64 characters per line (two screen lines) and number lines in multiples of 10.

The Dragon Users Club

To help everyone obtain the maximum enjoyment and benefit from their Dragon 32, the newsletter will pay specific attention to the formation of a Dragon Users Club.

A special feature in each magazine will be devoted to club news, giving details of specific events of interest to Dragon users, both at a local and national level.

Details of how to apply for membership of the Dragon Users Club will appear in the next edition of the newsletter which will be mailed free of charge to all existing users who have returned their guarantee cards for the 32 to Dragon Data Ltd.

Dragon Users

The newsletter will also report on the very latest developments happening within Dragon Data Ltd., the Welsh based manufacturing company that produces the highly successful Dragon 32.

1983 will see some exciting product news for Dragon users. Current plans include the introduction of a disk drive for the Dragon 32, scheduled to appear on the market in April.

For the latest news on software plans and for an up-to-date list of software currently available for the Dragon 32 see the back page of this newsletter.

Cassette Loading

Cassette loading and saving with the Dragon is generally straight forward but problems may be encountered with some Automatic Level Controls on cassette recorders.

If problems do occur with saving and loading your own programs onto cassette here are some tips that may be of use:

1. Check that cassette heads are clean, and clean the heads regularly.
2. Ensure that plastic tape header has been passed before saving.
3. If I/O errors are obtained and television is very close to Dragon, move Dragon further away from the television.
4. If problems still occur try removing input lead (MIC) whilst loading and output lead (EAR) whilst saving.
5. The problems caused by some Automatic Recording levels may be cured by using the following command (Put Motor on first)

   ```
   SOUND 120, 20:FOR N = 1 TO 100:NEXT C:SAVE "Program Name"
   ```

Programs f

DRAGON BYTES

Although DRAGON has 32,768 bytes of memory, which should be ample for most purposes, some of us always want to squeeze the last byte out of the machine. If you are embarking on one of those large multi-option programs, a few tips may help to put off the moment when the OM ERROR appears on your screen.

REM statements are very useful in developing and editing programs, but every letter takes 1 byte, and so these should be deleted if memory is running short. But be careful not to delete whole lines if the line numbers are used in GOTO or GOSUB statements.

Most spaces are unnecessary, and each takes 1 byte. But some of them must be kept in. Apart from the obvious ones inside quotes and in strings in DATA statements, a space is needed when a variable name is followed immediately by a word. In the following examples the spaces are necessary:

(a)FORI=XTO5 (b)IFX=YTHENSTOP (c)ONXGOTO1,2,3

These spaces can, in fact, be removed, but only by means of special programs, usually in machine code. The advantage of such programs is that you can keep a "readable" version and a "compact" version of your BASIC programs – without spaces and REM statements, it's not easy to modify them.

Use 1-character variable names in preference to 2-character ones, and never more than 2.

Finally, wherever possible write several statements on one line. Writing a statement on a new line takes 4 bytes more than adding to it an existing line.

With 32K to play with, you won't often need to worry about this sort of byte-budgeting, but it pays to develop economical habits, and it's more satisfying to program efficiently.

SKETCH PAD

This program uses the high-resolution graphics screen to draw pictures. The tiny cursor can be switched on and off using the space bar, set into motion using the arrow keys, and stopped using the full-stop. Shift with the arrow keys gives a 45° rotation. With a bit of practice you will soon learn to draw pictures with it. Then you might like to try modifying the program to use colours (on a lower resolution screen) and to make use of the PAINT command. If you have joysticks, you could convert the program to incorporate them.

```
 10 REM PROGRAM SKETCH
 20 REM A.D.MAYER, 1983.
 30 PMODE4,1:PCLS5:COLOR0,1:SCREEN1,0
 40 X=128:Y=96:PSET(X,Y,0):A=0:X1=0:Y1=0
 50 GOSUB190:IFZ<8 OR Z>95 THEN 50
 60 ONINT(Z/8)GOTO70,150,50,160,170,50,50,50,50,50,180
 70 ONZ-7 GOTO90,130,140,50,50,50,50,50
 80 Y1=1:X1=-1:GOTO50
 90 X1=-1:Y1=0:GOTO50
100 X1=1:Y1=-1:GOTO50
110 X1=0:Y1=-1:GOTO50
120 X1=1:Y1=1:GOTO50
130 X1=1:Y1=0:GOTO50
140 X1=0:Y1=1:GOTO50
150 IFZ><21 THEN 50 ELSE X1=-1:Y1=-1:GOTO50
160 IFZ><32 THEN 50 ELSE A=1-A:GOTO50
170 IFZ><46 THEN 50 ELSE X1=0:Y1=0:GOTO50
180 IFZ<91 THEN 50 ELSE ONZ-90GOTO80,50,120,110,100
190 GOSUB200:X$=INKEY$:IFX$=""THEN190ELSEZ=ASC(X$):RETURN
200 IFA=0THENPRESET(X,Y)
210 X=X+X1:Y=Y+Y1:X=X-INT(X/256):Y=Y-INT(Y/192)
220 IFX<0THENX=0
230 IFY<0THENY=0
240 PSET(X,Y,0):RETURN
```

TYPING GAME

This is a little program to help you find your way around the keyboard. It uses INKEY$ to check when you have typed the letter and the TIMER to time your reaction.

```
  1 ' :: TYPING GAME ::
  2 ' MAVIS PEARSON. 1982
 10 CLS:PRINT@138,"TYPING GAME":PRINT:T2=0
 20 PRINT"THIS PROGRAM ASKS YOU TO FIND"
 25 PRINT"A LETTER ON THE KEYBOARD AND"
 30 PRINT"THEN TYPE IT.THEN IT WILL TELL"
 35 PRINT"YOU HOW LONG YOU TOOK."
 40 FOR I=1 TO 2000:NEXT:CLS
 50 FOR K=1 TO 10:C=RND(26)+96:TIMER=0
 60 PRINT@192,"TYPE THE LETTER ";CHR$(C):PRINT:SOUND 100,1
 70 I$=INKEY$:IF I$="" THEN 70
 80 PRINT,I$:PRINT:PRINT
 90 C=C-32:IF I$=CHR$(C) THEN 110
 95 PRINT" BUTTERFINGERS !! PENALTY 5 SECS"
100 TIMER=TIMER+252
110 T=TIMER
120 PRINT "REACTION TIME:";INT(T/51*1000)/1000;"  SECONDS"
130 FOR I=1 TO 2000:NEXT:PRINT:T2=T2+T:CLS:NEXT K
140 PRINT"AVERAGE REACTION TIME OVER"
150 PRINT"TEN TURNS";INT(T2/51*100)/1000;"  SEC."
160 PRINT"HOW ABOUT ANOTHER GO? Y/N"
170 Y$=INKEY$:IF Y$="" THEN 170:PRINT:PRINT
180 T2=0:CLS:FORI=1 TO 1000: NEXT
190 IF Y$="Y"THEN CLS:GOTO50
200 END
```

If you wish to learn to type, there is a Dragon tape available called "Typing Tutor".

WHAT ARE HEXADECIMAL NUMBERS?

At the heart of your Dragon is a microprocessor that operates on storage locations called BYTES. Each byte consists of 8 locations, which can be either 1 or 0, forming a binary number between decimal 0 and decimal 255. Binary numbers are very difficult to deal with without making mistakes, and the 256 numbers can be represented by two digits, each of which can be the symbols 0,1,2,3,4,5,6,7,8,9, A (for 10), B (for 11), C (for 12), D (for 13), E (for 14) and F (for 15). Your Dragon has a built in function HEX$(N) which converts the decimal number N to its hexadecimal equivalent. But you might like to follow through the small program which performs the same operation.

```
 10 INPUT"ENTER A POSITIVE INTEGER";N
 20 H$="":X=N
 30 Y=INT(X/16)
 40 RE=X-16*Y:GOSUB100:H$=A$+H$; IF Y=0 THEN 60
 50 X=Y:GOTO30
 60 PRINT"HEX(";N;")=";H$:GOTO10
100 IF RE<10 THEN A$=MID$(STR$(RE),2):RETURN
110 A$=CHR$(RE+55):RETURN
```

s for your 32

46016

48024

MACHINE CODE CORNER

Machine code is the language your Dragon understands. It can only communicate in languages like BASIC by means of an "interpreter"; and translating languages takes time – even for a Dragon. So if you write your instructions in machine code, they will be carried out faster, and in some cases you will be able to do things which are not possible in BASIC.

At first sight, machine code doesn't look much like a language at all. It has no words – just a series of numbers or "bytes". Each byte lies between 0 and 255 (8 bits) and represents an instruction, or part of an instruction. The Dragon executes these instructions in order, in much the same way as a BASIC program is executed. A long and complex machine code program is a permanent feature of your Dragon – it is the BASIC interpreter, and is held in ROM (Read-Only Memory) between decimal addresses 32768 and 49151, hexadecimal 8000-BFFF (for more information on Hexadecimal numbers see opposite.) To look at a small part of this program, try running the following Basic program:

```
10 FOR I = 46080 TO 46147
20 PRINT PEEK(I);NEXT I
```

Then to find out what this machine code program does, type

```
EXEC46080
```

(All commands of this sort must, of course, be followed by "ENTER".)

By the way, it's not a good idea to EXEC46080 if you have an important program in memory, since one of the results is the same as typing "NEW"!

This area of Dragon's memory is ROM, and so it can't be altered. When we write our own machine code programs, we must locate them in RAM (Random Access Memory), between decimal addresses 0 and 32767 (hex 0-7FFF). All of these can be altered using the POKE command, but some of them are used by the system for special purposes. Of particular interest to us is the range 1024-1535 (hex 400-5FF) which is the Text Screen Memory. Altering these locations causes characters to appear on the Text Screen.

A good place to locate machine code programs is in the highest RAM locations (say 32000-32767) since this region can be "protected" from BASIC using the CLEAR command.

The easiest way to produce machine code is by writing in "assembly language", which is a mnemonic representation of the code, and using an assembler to assemble the machine code. Two Editor-Assembler packages are available from Dragon Data Limited. An assembler is essential equipment for writing anything but the simplest machine code routines.

The following is an example of a simple program to fill the Text Screen with the character whose ASCII code is stored in the highest RAM address (hex 7FFF).

Assembly Language	Description	Machine Code
1 LDA $7FFF	(Load A)	182,127,255
2 LDX $400	(Load X)	142, 4, 0
3 LOOP STA ,X+	(Store A)	167,128
4 CMPX $600	(Compare X)	140, 6, 0
5 BNE LOOP	(Branch if not equal)	38,249
6 RTS	(Return from subroutine)	57

Line 1 loads register A with the contents of memory 32767 (hex 7FFF). Line 2 loads register X with the value hex 400 (the address of the top left corner of the Text Screen). Line 3 stores A in the memory indicated by X and increments X by 1. Line 4 compares X with the value hex 600 and line 5 causes a branch to line 3 if X is not equal to hex 600. Line 6 returns control to the calling program.

As you can see statements in this program take 1,2 or 3 bytes when translated into machine code. The whole program becomes

```
182 127 255 142 4 0 167 128 140 6 0 38 249 57
```

To load this program, without using an assembler, you must write a short BASIC program:

```
10 CLEAR 200,32000
20 FOR I = Z TO 13: READ J: POKE I + 32000,J: NEXT I
30 DATA 182,127,255,142,4,0,167,128,140,6,0,38,249,57
```

This will "POKE" the machine code directly into the memory, starting at 32000, having first protected that area with the CLEAR command. Before running, it is wise to check very carefully that it has been copied correctly. Unlike BASIC programs, machine code routines will not return control to you with an "error" message if you make a mistake. Sometimes the Dragon will just refuse to respond no matter what keys you press. Should this happen, control *may* be restored by pressing the RESET button, but it is generally best to switch off and start again. but don't worry – you can't damage your Dragon, no matter what you POKE into its memories; it will always put itself right if you switch off and start again. So you can "experiment" as much as you like.

Now we have written and loaded our machine code program, it is time to see what it can do. It is best to run it under the control of a BASIC program, such as the following:

```
10 CLEAR 200,32000: CLS
20 X$ = INKEY$: IF X$ = "" THEN 20
30 POKE32767,ASC(X$)
40 EXEC32000: GO TO 20
```

Run this program, and try pressing different keys. the screen should fill with the relevant character instantaneously. You may like to try to achieve this result using BASIC only, and see how fast you can fill the screen. The difference between letters and other symbols is interesting. Try pressing (Shift 0) and see what effect this has on subsequent keys.

Now obtain control by pressing the BREAK key, and change line 30 to read **30 POKE32767,ASC(X$) – 64**

and run this new program. the symbol keys will now give a "syntax error", but the letter keys still work – in reverse screen.

Now replace line 30 with **30 POKE32767,ASC(X$) + 128**

The result is an interesting display of graphics.

If you would like to save your machine code on tape, the command to use is CSAVEM"SCREEN",32000,32013,13. SCREEN can be replaced by any filename you choose; the first two numbers are the addresses of the beginning and end of the machine code; and the third number is the difference between them. All of the parameters are necessary. The machine code may now be loaded from tape using CLOADM"SCREEN". The BASIC program can be saved in the usual way (CSAVE).

DRAGON SOFTWARE

The range of software available for the Dragon 32 will be increasing all the time. More games on cassette and cartridge are scheduled for release over the next few weeks, including a series of new cassettes for "adventure" fans.

On the subject of adventure games, Madness and the Minotaur seems to be living up to its name. As we have had a number of enquiries concerning the elusive "Mushroom" we are taking the opportunity to give you a few more clues.

1 The mushroom can be found two floors down from the entry floor.
2 To descend you will require the lamp. To switch the lamp on, enter the command "LAMP ON".
3 On the floor where the mushroom is to be found, you stand a good chance of getting lost in a seemingly endless maze. Going as far South and as far East as you can, should get you out of that.

The first three of a series of ten educational cassettes should be available by the time you receive this newsletter. The other seven will become available over the next six months. The programs are designed to assist the numeracy and literacy development of four to twelve year olds. The first titles are: –

1 Hide and Seek	Designed to encourage short term memory and aid development of early reading skills.
2 Number Puzzler	Designed to improve children's ability to do mental arithmetic.
3 Number Gulper	Another cassette designed to improve arithmetic which children will enjoy using.

SOFTWARE AVAILABLE FOR THE DRAGON 32

A.0500	**DRAGON SELECTION ONE**	Four games for the younger user. Written in BASIC, they can be listed and edited.
A.0501	**DRAGON SELECTION TWO**	Collection of utilities. Create your own data base, write your own tunes.
A.0502	**QUEST**	Adventure game in a medieval setting. Defeat Moorlock, master of the dark castle.
A.0503	**MADNESS AND THE MINOTAUR**	A real-time adult adventure game.
A.0504	**PERSONAL FINANCE**	Keep track of family finances.
A.0505	**GRAPHIC ANIMATOR**	Create simple cartoons on the screen and animate them by flipping through the pages. Joysticks required.
A.0506	**COMPUTAVOICE**	Your Dragon will talk with this speech synthesizer.
A.0507	**EXAMPLES FROM THE MANUAL**	30 examples from the programming manual.
A.0508	**CALIXTO ISLAND**	An adventure game. Return the hidden treasure to its rightful place.
A.0509	**BLACK SANCTUM**	An adventure game. Overcome the forces of black magic.
A.0512	**TYPING TUTOR**	Improve your speed and accuracy.
A.0513	**DRAGON MOUNTAIN**	An adventure game. Defeat the guardians of the treasure hidden in the mountain.
A.0514	**FLAG**	Race your opponent through a constantly changing maze to the final flag. Joysticks required.
A.0518	**EL DIABLERO**	An adventure game set in the desert.
A.0100	**BERSERK**	A challenging shooting game, based on the popular arcade game, one or two players. A high resolution game in black and white. Joysticks required.
A.0101	**METEROROIDS**	Guide your ship through treacherous asteroid belt. A game requiring skill, fast reactions and concentration. A high resolution game in black and white. Joysticks optional.
A.0102	**COSMIC INVADERS**	Dragon version of the famous arcade game.
A.0103	**GHOST ATTACK**	Maze game for one player. Joysticks required.
A.0104	**CAVE HUNTER**	Descend into the maze of caves in search of gold. Joysticks required.
A.0106	**STARSHIP CHAMELEON**	Protect your planet from the attacks of the Gabulators. High quality arcade game with superb graphics and sound. Joysticks required.
A.0107	**ASTROBLAST**	Defend your ship against waves of attackers. A high resolution game in black and white. Joysticks required.
A.0108	**CHESS**	Nine levels of play, from beginner to master.
A.0111	**RAIL RUNNER**	Move Bill Switchman across the tracks, avoiding trains, to rescue Herman Hobo.

Published by: Welbeck Public Relations, 2 Endell Street, London WC2. Printed by: Drydens Printers Ltd., Brent Crescent, North Circular Road, London NW10.