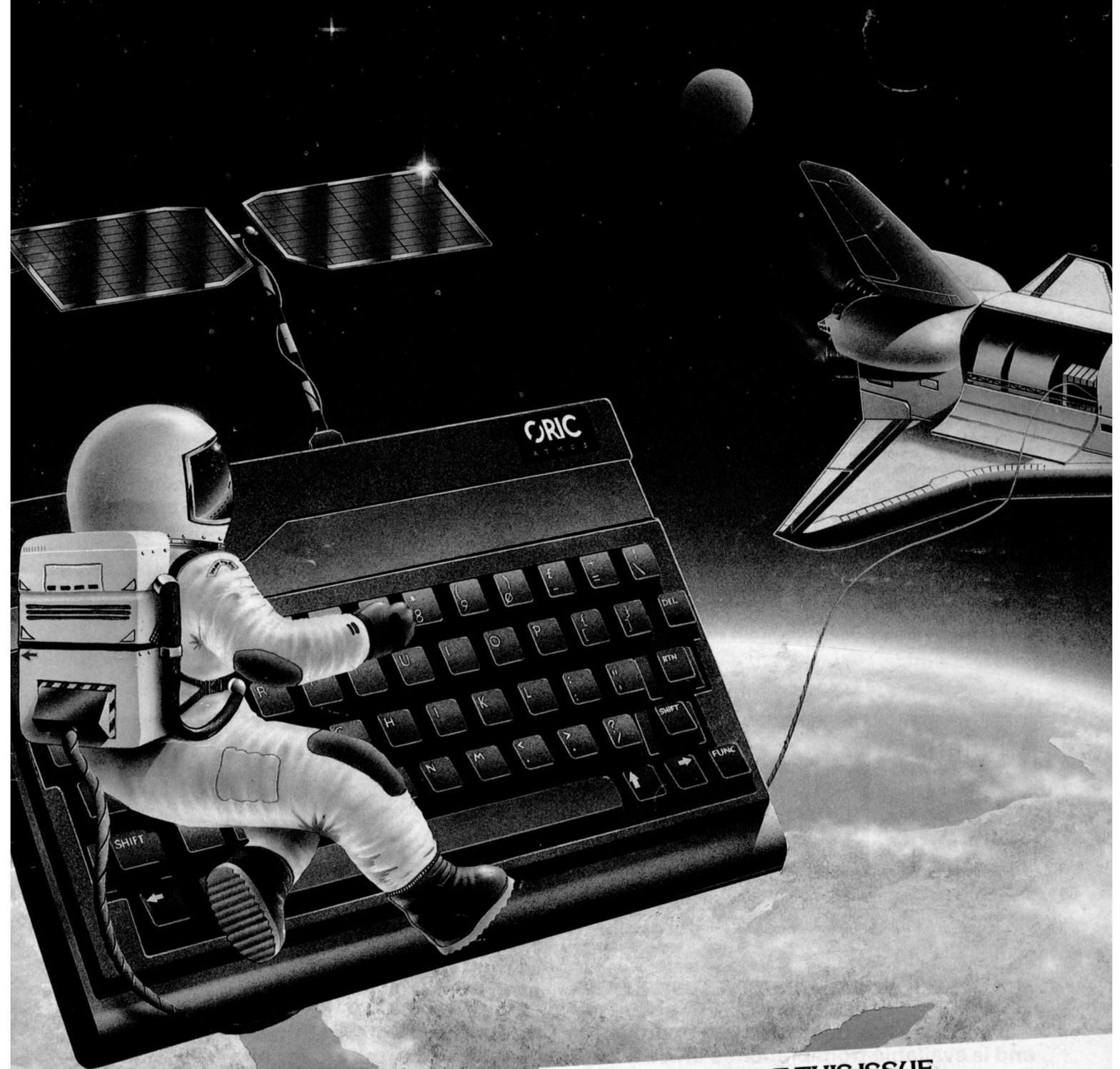


ORIC OWNER

ISSUE 6

FEB/MAR '84



INSIDE THIS ISSUE

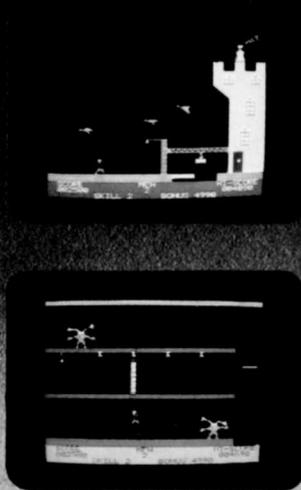
First look at the Atmos

Forth: Screen dump

Parallel to RS232C Interface

Introduction to Machine Code

IJK Software and the ORIC-1 Micro bring you more exciting arcade action from the Xenon series...



From the author of the highly acclaimed Xenon-1 comes the second in the Xenon series. This game continues the high standard of excellence set by its predecessor. The amazing super high resolution graphics make this 100% machine code arcade game a must for your collection.

Following their defeat at the hands of the Xenon fleet the Zоргons have captured the Xenon princess Roz, and have imprisoned her in their castle. You are commissioned to rescue her by scouring the four corners of the Zоргon Empire to capture the magic stones. These stones, guarded by the Quadnogs, Terrapods and many other strange beasts, are needed to bridge the bottomless chasm surrounding the castle, enabling you to achieve your goal.

Each one of the many varied stages in this scintillating mission will test your arcade ability as never before.

FEATURES INCLUDE:-

- 100% MACHINE CODE
- SUPERB SOUND EFFECTS
- SUPER-SMOOTH HI-RES GRAPHICS
- MULTI-STAGE ARCADE ACTION
- SKILL LEVELS
- HALL OF FAME

For the 48K ORIC-1 £8.50 inc.

OTHER TITLES IN THE IJK SOFTWARE ORIC-1 RANGE...



XENON-1
100% M/C arcade
game for the
48K ORIC-1
£8.50 inc.



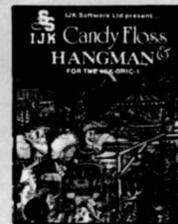
INVADERS
Machine code
arcade game for
16K & 48K ORIC-1
£7.50 inc.



FANTASY QUEST
Intriguing
adventure for
48K ORIC-1
£6.50 inc.



REVERSE
Superb board
game for
48K ORIC-1
£6.50 inc.



**CANDYFLOSS &
HANGMAN**
Two top
educational
programs for
48K ORIC-1



**3D MAZE &
BREAKOUT**
Arcade action for
48K ORIC-1
£7.50 inc.

ALL PRICES FULLY INCLUSIVE OF VAT and P&P - NO MORE TO PAY

Our software has been officially approved by **Oric Products International Ltd.**, and is available from branches of **W.H. Smiths, Laskys, Computers for all, Zappo, Boots, John Menzies, The Spectrum Chain**, many other leading department stores, hundreds of independent dealers nationwide, and in **23 countries across the world**. You can also order direct from us - all advertised software is in stock now and will be despatched within 48 hours of receipt of order.



**IJK
Software
Limited**



24 HOUR ANSAFONE

Unit 3c, Moorfields, Moor Park Avenue,
Bispham, Blackpool, Lancs FY2 0JY
Telephone (0253) 55282

WARNING: All software sold subject to IJK Software's standard conditions of sale and terms of trade, copies available on request.



ORIC OWNER

Issue 6 Feb/Mar

Features

	page
Practical Machine Code (Part 1)	8-12
Getting the Most from Oric-Base	14-15
Parallel to RS232C Interface	18-21
Low Resolution Graphics (from the Oric Handbook)	33-36

Regulars

Editor's Comment	4
Captain Tanex	16
Disaster Area	17
1:0 Page	45
Oric Quickies	49-53

News

News Brief	5-6
New Products Review	25-26

Software

Software Scan	
Telephone Directory 13 and 31	
And Forthly . . .	22-24
Chaser	37-38
Battleships	39-41
Doggy Bone	42-44
Spelling	46-48

Oric Owner Magazine

Editor: Paul B. Kaufman

Graphics & Feature Editor:

Carolyn Groeneveld

Administration: Carolyn

Groeneveld

Technical Consultant: Dr Paul Johnson

Printers: Heffers Printers Ltd, King's Hedges Road, Cambridge.

Photographic Equipment: Minolta

Oric Owner is published at bi-monthly intervals by Tansoft Ltd, Reg. No. 1632070

Units 1 & 2,

Cambridge Techno Park,

Newmarket Road, Cambridge.

Advertising Rates are available on request.

© Tansoft Ltd 1984.



New Products Review

A first look at the incredible new Oric Atmos



And Forthly . . . Screen Dump

A useful utility for all Forth users.



Getting the most from Oric-Base

More help for this popular program.



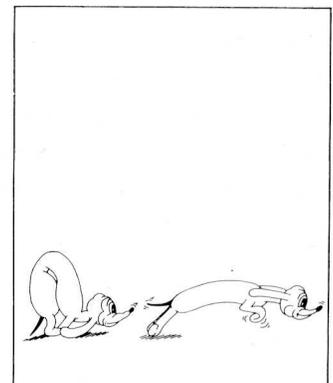
Telephone Directory

This is a program to store all those important telephone numbers that you keep forgetting.



Captain Tanex

Another Entertaining Adventure.



Chaser

Here's your chance to try and catch the computer out!

Editor's Comment

By Paul B. Kaufman



When I mentioned in issue 4 that I was looking for software reviewers I expected to be completely underwhelmed by half a dozen or so replies but much to my surprise I received dozens of them. Letters are now going out to all those who wrote although obviously not everyone can be used. So many thanks to all of you and if you are a specialist in one particular area I would still be delighted to hear from you.

A number of people have complained that listings printed in the magazine have faults or errors in them. We are taking more care now to spot them before they get into print but

one of the problems is that when people send their programs to us on cassette, they often only record a fast version and as you know the Oric is very sensitive to tape problems at the fast speed. So if you are one of our contributors please, please send us your programs at both fast & slow speeds for everyone's peace of mind.

Due to popular request I have printed a picture of my cat, Trillian, to replace the usual grotty picture of myself. In the next issue will be a picture of a chicken I saw on holiday.

If you have just received your new Oric for Christmas I would

like to welcome you to the Oric Owner Magazine and am sure that you will find it invaluable when you are stuck in one of those all night programming sessions. And don't forget that we always are on the lookout for good quality programs and articles to print.

Apologies for the slight delay in sending out the magazines but we have been converting our subscription system to run on our new Apricot Computer and it was a bit more difficult than we first expected. By the way, don't forget that your subscription may need renewing next issue – we'll let you know.

News Brief

Oric Launch Atmos at Which Computer? Show

Stand No: 2403
Hall 3A NEC
Jan. 17th 1984



Launched at the Which Computer Show, and in full production as from Jan 16 '84, is Atmos, a new 48K British microcomputer from Oric Products International.

Superseding the Oric 1, of which more than 160,000 were shipped during 1983, the Atmos incorporates a number of new features and refinements, the most obvious being a professional full-pitch typewriter keyboard, and a re-styled case in black and red. The Oric four colour printer and new 320K byte 3" micro disc drives have also been re-styled in the same colour scheme.

Although Basic programs are totally software compatible with the Oric I, the Atmos has a new ROM operating system which, claim Oric, greatly enhances the performance over the original Oric 1. See our full review further on.

New Software

An exciting new range of software for the ORIC-1 is to be launched on 1st March.

It is called 'Softbacks'.

The name comes from the idea that the computer will increasingly take over from books as the medium of information storage in the home. A computer can store as much information as a book and has the added dimension of intelligent presentation. Where you once bought a notebook to write in or a text book to learn from, you will in future probably buy a piece of software which will enable your computer to do the job.

The three programs available initially are good examples of how the computer can add the extra dimensions.

The first is ACCOUNT BOOK, a personal finance program. You put the figures in, the computer adds them up and presents them in the form of summaries that show at a glance how you stand financially. Features include Account Definition, Budgeting, Standing Orders and File Handling. It produces a comprehensive range of reports on printer or screen.

The other two are educational programs which up to now have been hard to find for the ORIC.

STORY BOOK takes any piece of text and replaces the words by dashes. The aim is to reconstruct the story and as you guess the words they are put back to their places on the screen. It stimulates early readers and foreign language students in a way the printed page cannot hope to match.

PICTURE BOOK is designed with young children in mind. They can build up a picture of familiar objects – if they can spell the names! It comes with a built-in picture dictionary of key words for early learning, and extra words can be added.

Further information may be obtained from Softbacks, PO Box 257, Watford WD1 3LQ.

STOP PRESS

Computer World opens up in Beckenham this week to sell home computers. Initially it will stock more than 10 models of home micro and others will be added soon – ZX81 Spectrum 48K Dragon 32 Oric 1-16/48K Vic 20 Commodore 64 CLG SORD M5 BBC B Lynx 96K. Other models can be ordered. Besides this is as wide a range of software, (even for machines not stocked) as is found in shops in central London – hundreds of cassettes including education and business software.

News Brief

Tansoft convert

Tansoft have completed the conversion of their software titles to run on the Oric Atmos. They all retain compatibility with the Oric-1. Customers ordering software are requested to state which machine they have.

PSS convert

With the imminent release of the new Oric Computer PSS have now completed conversion of all existing Oric programs. Tapes will be sold containing versions for both Orics.

PSS tapes will now contain versions of the game for both Orics and this will continue until further notice. Owners of the Oric 1 will also be supported on all new releases.

For further details contact John Fletcher at PSS on:- Coventry (0203) 81346.

Graphics & Programming for the Oric

A course is being offered by the University of Birmingham entitled, 'Graphics and Programming for the Oric'.

The course will include the basic concepts of computer graphics – namely windowing, drawing of lines and curves, transformations (shifting, scaling and rotating) and a discussion of the main concepts of G.K.S. 'Graphical Kernel System' which is soon to become the new international standard in two-dimensional graphics.

More information from: Mrs S. Laflin-Baker, Aston Webb Building, Lecture Room 7, University of Birmingham.

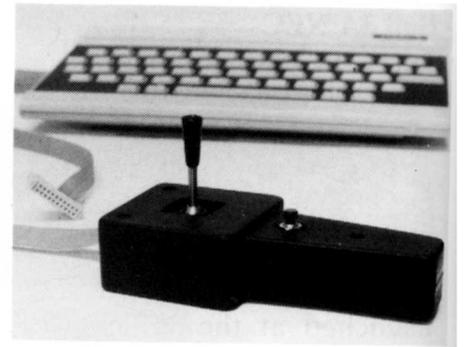
'Oric-1 Machine Code' by Bruce Smith

Shiva Publishing have published a paperback by Bruce Smith called 'Oric-1 Machine Code', for those Oric owners with some knowledge of BASIC.

The book teaches the reader how to program the Oric-1 in machine code, which allows programs to be RUN more quickly and special effects to be created. It includes information on Hex and BCD, registers, absolute and indirect addressing, macros, the stack and how to use MOS routines.

It will cost £6.95 and will be released in March 1984.

Joystick has its own interface



Pennant announce a self-centring joystick which plugs straight into the printer port of the ORIC-1.

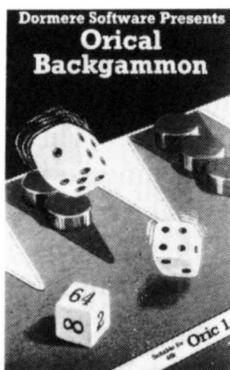
It contains its own interface which gives eight-way movement and high speed firing with full simultaneous sound effects. A sturdy micro-switch mechanism is used for high sensitivity and provision is made for the connection of a second joystick or printer if required.

Priced at £14.85 it is supplied with a demonstration program.

Pennant,
29 Hounds Road,
Chipping Sodbury,
Bristol

Software Scan

Backgammon – Dormere Software Price £7.95



This is a package from a Company I have not heard of before. I farmed the game out to several people who are better players than myself and they concluded that it actually plays quite a strong game. On the graphics side, I felt that the board could be improved a little but with these sort of games, the actual playability is far more important. Not too bad at all!

By Paul B. Kaufman

Franklin's Tomb – Salamander Software Price – £9.95

Salamander's 'Franklin's Tomb' is a text only adventure. The playing instructions and illustrations of the locations are in a



well-produced booklet accompanying the tape.

An unusually long wait after the opening credits displays the basic plot – pressing the wrong key and you are straight into the game. Thought has gone into the screen display. On the left is your location and on the right is an inventory of what you are carrying. These do not scroll and are only up-dated when you pick up something or move to a different location. The bottom half of the screen is used for your inputs and the computer's replies.

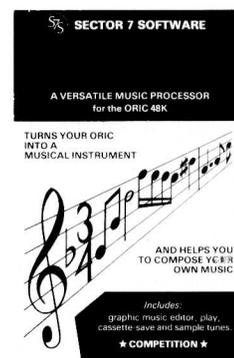
The game is a nice mixture of humour and puzzles with only one location so far that you cannot get out of.

If forms part of a trilogy but each can be played on its own.

At £9.95 it's a bit on the expensive side, especially as once you have solved an adventure, it is no longer playable. If you like this sort of programme, buy it – Well thought out and entertaining.

By P. A. Winter

Composer – Sector 7 Software Price £6.50



This package allows you to compose three part music on the Oric. It allows you complete control over virtually every aspect of music generation.

You enter your piece of music on staves which are displayed on the screen. A comprehensive 'music' editor allows you to insert or delete bars or move them around.

There are several sample tunes supplied on the cassette which impressed me greatly.

Sector 7 Software are offering prizes to the best compositions produced with the program. If you are a music enthusiast then I would strongly recommend this package.

By Paul B. Kaufman

Practical Machine Code (Part 1)

By D. Sinfield

As a Microtan owner I was forced into learning 6502 Machine Code by lack of funds to buy the extensions necessary for high level languages. When I switched to the Oric I was struck by the sluggishness of the BASIC compared to machine code programs. This really came home when I read a letter in Oric Owner from someone asking about fast sideways scrolling. If there ever was an example of machine code being worth learning this is it. Most of the real faint-hearted probably stopped at the first mention of machine code but I hope to prove that it's not only the veteran m/c games writers (most of them half my age) who can use machine language.

Working with the screen memory is a good place to start because the first thing it is important to understand is that at the machine code level we are really dealing with the shuffling about of bytes of memory and on the screen we can see this happening.

There are mountains of books and articles on 6502 code but most are processor oriented and tell all about what each instruction does but not how to put them together. This is an attempt to develop a routine

whilst explaining the steps along the way.

The first thing to do is to set a goal so that you know that the program is finished. Otherwise you'll be adding bits and pieces to the routine for ever. The first goal that we can set ourselves is to produce a program that scrolls right with full wrap around on the first line of the screen. Now we know where we're going we can set off.

I use flow charts a lot and the first process is to do a macro-algorithm, which is just a flow-chart that isn't too detailed but outlines the processes. Figure 1 shows the sort of thing. You'll notice that the flowchart is linear and that's the way machine language works so flowcharting like this seems sensible. As a matter of style I like one start (at the beginning) and one finish (at the end). Sometimes this adds a couple of bytes and sometimes it doesn't work out at all.

Now come away from the keyboard – I know it makes a nice noise and puts up neat typing on the screen but before getting to the point of entering the program there's lots of working out to do first! The savings made in the speed at which the program runs are to some extent paid for

at the stage of program development and debugging.

As a middle step between us and the machine we are going to use Assembly code. This simply takes the machine code instructions and makes them a little more human. There is a list of 6502 instructions and their mnemonics in appendix K of the Oric manual. Don't worry too much about them now – we'll be looking at each one as it is used.

Before starting the program let's look at the Oric's screen memory. It can be represented by a matrix like Figure 2. To put a character in a given screen position we put its ASCII code in the appropriate memory location. This can be done in BASIC using POKE – try POKE#BFDF,48. Given that 48 is the decimal ASCII code for 0 the result should be predictable. To point to these screen locations the program is going to use an addressing mode called indirect, which means that the actual location is not in the program – it is in fact in page zero memory i.e. somewhere between 0000 and 00FF. The program is given the location of this pointer. The first thing our program has to do is set up these zero page addresses to point to

Regulars

the top left hand corner of the screen. To keep things uncomplicated by attributes we will keep the paper and ink columns untouched. Our pointer wants to indicate the memory location BBAA. From now on, by the way, all numbers will be in hexadecimal unless otherwise stated.

The Oric is BASIC oriented and will go back to the interpreter when the present program is over. To make sure that our program can coexist with the interpreter it is necessary to use some zero page locations that aren't going to be used in BASIC. It so happens that the lowest locations are free and this program is going to use 0000 and 0001. The 6502 requires that the low byte of the address pointer (the last two digits) must go into the lowest of the zero page locations.

The first task then is to get AA into location 0000. Look at the program in Figure 3. Only worry about the first two columns for now. Line one *LoaDs* the Accumulator with the value AA. The accumulator is a byte of RAM actually inside the 6502 and as you'll see most transactions pass through it. We show that the actual value AA is to go into the accumulator by using the hash mark (#) between the instruction (LDA) and the operand (AA). Line two puts the contents of the accumulator at location 00 using the instruction *STore Acc*. Notice that there is no hash mark here because 00 isn't a value, its a memory location. The next two lines use the

same instructions but different operands to load the value BB into memory location 0001. We can trace the values in each relevant location like this: † n/k = Not known

After Line	Accumulator	0000	0001
1	AA	n/k	n/k
2	AA	AA	n/k
3	BB	AA	n/k
4	BB	AA	BB

The equivalent in BASIC would be:

```
POKE 0, 170:POKE 1,187
```

The pointer in zero page, remember, points to the top left of the screen and to scroll right we have to start on the right (think about it). The program is going to use the Y register both to change the address pointed to and count the number of characters in a line. First Y has to be assigned a starting value that represents the number of characters on a line, ignoring the two attribute columns. The manual says that this is 37 in decimal which is 25 in hex. To load this value into Y the instruction *LoaD Y* is used, again in the immediate mode, in line 5 of the program.

Now, well and truly initialised, we can get on with moving things. As we are aiming at wraparound the right hand character has to be stored until a space has been created for it on the left. The first task is to get into the accumulator with a load A instruction. Look at line 6 of the program and try not to panic. The line looks complex but it isn't really. LDA we know, the operand in brackets

means that this isn't an address but an address of an address – the one we've just set up – and the Y shows that the value of the Y register is to be added to the address pointer to arrive at the correct address of the value to be obtained. Taking line 6 as an example we get: The address pointed to by the operand is 00. At 00 there is the low order byte of the address (AA) The next highest location holds the high order byte (01=BB) Put these together in the right order (BBAA) And add the value of the Y register (25 + BBAA = BBCF) Now put the value held at this address in the accumulator.

If you consult figure 2 again you'll see that the address pointed to sure enough is the top line far right character. So after line 5 the accumulator will hold the ASCII code of this character. The next line (7) puts the ASCII code in a temporary store called the stack. The mnemonic PHA stands for *PusH Acc*. and all it is necessary to know at the moment is that if you push more onto the stack than you pull off it you'll lose the return address and end up with a crash.

Having dealt with the character on the far right we want the next one to the left of it. To move the pointer left it has to be decreased by one. This is done by line 8. Try reducing Y by one and working through the steps in the paragraph before last and you'll see that the pointer will be indicating the next left character. There is an instruction to subtract one from Y called *DE-*

Features

crement *Y*. It takes up only one byte as does *PHA* because all the addresses, amounts etc. are *implied* by the instruction. This gives the name of yet another addressing mode.

Line 9 is familiar. Line 10 moves the pointer right by *IN*crementing *Y* and line 11 puts the character code that's held in the accumulator back on the screen one position to the right of its previous position. Line 12 puts the pointer back to where it started after line 8.

After each character has been done the program checks that the end of the line hasn't been reached in line 13. All the screen positions on a line will have been done when decrementing *Y* in line 12 results in zero. This is tested using the Branch if Not Equal (in this case read *zEro* for equal). This instruction is self explanatory but the operand part of line 13 is a little obscure. In its branch instructions the 6502 uses relative (yet another mode) addressing. Instead of telling the processor the address of the next instruction to be done if the condition is true the operand represents a number of forward or backward jumps that have to be made to arrive at the destination instruction. To calculate branch offsets is easy following the steps below:

Forward Branches

- 1 Add two to the address of the branch instruction.
- 2 Subtract the above from the destination address.

Backward Branches

- 1 Subtract the destination

address from the branch instruction address.

- 2 Subtract the result from *FE*.

Don't forget that all the arithmetic is in hex. The above is the didactic method it tells you what to do, for details of why you do it and the intricacies of two's complement refer to just about any book or article on machine code or binary arithmetic. Apply the backward branch method to destination *OD* and branch address 14 the result should be *F7* – the operand in line 13. The fact that *Oric* has a hex mode in *BASIC* means that it's easy enough to write a little program to work these out for you. The advantage of relative addressing is that the program can be located anywhere in memory without having to alter the branches.

Because of the branch in line 13 execution of line 14 only takes place if *Y* is zero and the whole line has been moved and the ASCII code for the character that was on the far right can be put on the far left. It is retrieved from the stack using *PuLl Acc.* As *Y* is zero the pointer must be indicating the far left column and *STore Acc. (00)*, *Y* does the trick.

The whole line has now been moved and the *ReTurn* from Subroutine in line 16 returns control to *BASIC*.

Unless you're lucky enough to have an assembler the work's not over yet. Assembly code is too human for the machine and it has to be coded. This can be done from appendix K of the manual but I find that the table

is that in fig 4. If you run down the code in column 4 of figure 3 comparing it with the assembly code in column 2 you should be able to see how the machine code is finally arrived at. The thing to remember is that when it's in RAM instructions look like ordinary hex numbers and it is the instruction itself that tells the machine whether to expect an operand and how many bytes long this has got to be. Failing to include the correct length operand means the program will get out of step and will certainly crash – probably fatally.

Now you can get on to the keyboard and enter the code. There are several ways of doing this the one included in the manual consists of turning the code into data statements but for our purposes I have written a short *BASIC* program to do the job. It does some validity checks but beware of entering the letter *O* instead of zero. The best way to check the code has been entered correctly is to disassemble it. The routine will go anywhere there is spare protected RAM. I put it at 9800 onwards and protected it with *GRAB: HIMEM#97FF*. If you do the same it can be saved using:

```
CSAVE"MOVER",  
A#9800,E#9819  
but don't blink or you'll miss it!
```

To run the routine make sure that there is something on the top line of the display and that the current print position is not so near the bottom that the top line will be scrolled off and

```
CALL#9800
```

Features

everything in the top line should have moved one place right. Now try this:

```
FOR N=1 TO 500:
CALL#9800:NEXT
```

the top line will start whizzing round at a mind-boggling rate. You might like to try it again with a short WAIT to slow things up.

This routine on its own is fairly trivial but it can be applied to any line by changing the pointer.

This can be done by altering the initialising routine or POKEing the correct values to the pointer and then calling the program from #9808 like this:

```
A=#FA: B=#BB
POKE 0,A:POKE 1,B:
CALL#9808
or DOKE 0#BBFA
```

should move the third line down across one position. Beware of getting into the two protected columns or into memory that has nothing to do with the screen.

Now try changing line 6 to:

```
Assembly machine code
LDA 20 A9 20
```

Given that 20 is the ASCII code for space, can you see what's going to happen? CALL the program a few times and see if you got the expected result.

Experiment with the program, very few changes will cause a leftwise scroll. Remember that the worst that can happen is a fatal crash which means that you'll have to power down and start again.

Fig. 1.

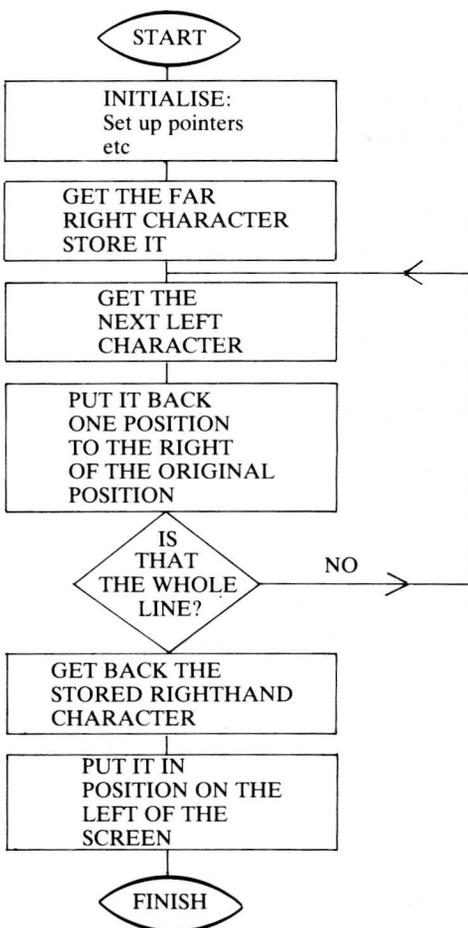


Fig. 2.

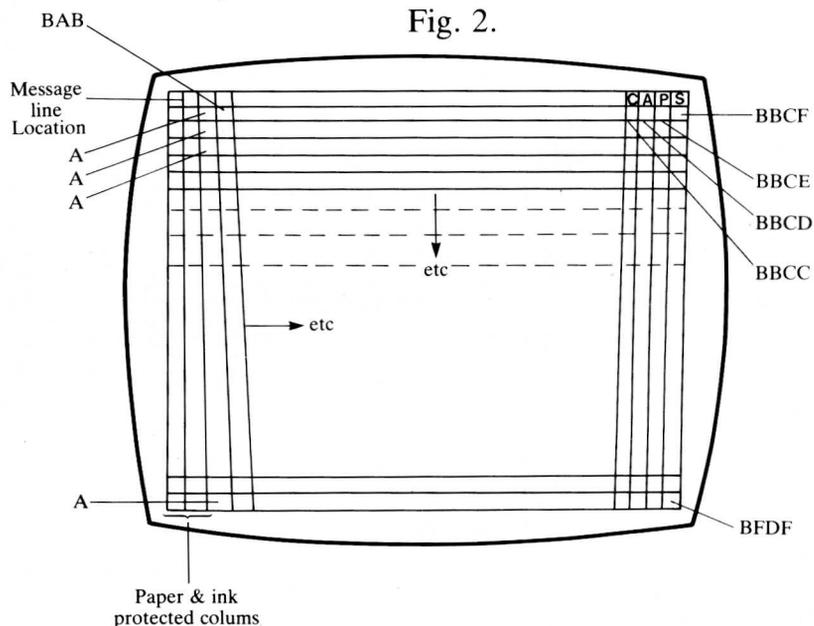


Fig. 3.

Line	Assembly	Address	Code	Remarks
1	LDA#AA	9800	A9 AA	
2	STA 00	02	85 00	
				Initialise set up pointer
3	LDA#BB	04	A9 BB	
4	STA 01	06	85 01	
5	LDY#25	08	A0 25	Point to right of screen
6	LDA (00), Y	0A	B1 00	Get the character pointed to
7	PHA	0C	48	And store it
8	DEY	0D	88	Move pointer left
9	LDA (00), Y	0E	B1 00	Get character
10	INY	10	C8	Move pointer right
11	STA (00), Y	11	91 00	Put character there
12	DEY	13	88	Move pointer left
13	BNE F7	14	D0 F7	Have we done whole line
14	PLA	16	68	Y=0: Line done. Get stored character
15	STA (00), Y	17	91 00	Put it on far left
16	RTS	19	60	Go back to basic

Features

Fig. 4.

	IMMEDIATE 2 BYTES	ZERO PAGE 2 BYTES	ZERO PAGE, X 2 BYTES	ABSOLUTE 3 BYTES	ABS. X 3 BYTES	ABS. Y 3 BYTES	(IND. X) 2 BYTES	(IND. Y) 2 BYTES	IMPLIED 1 BYTE	ACCUMULATOR 1 BYTE	RELATIVE 2 BYTES	
ADC	69	65	75	6D	7D	79	61	71				ADD with Carry
AND	29	25	35	2D	3D	39	21	31				AND
ASL		06	16	0E	1E					0A		Arithmetic Shift Left
BCC											90	Branch if Carry Clear
BCS											B0	Branch if Carry Set
BEQ											F0	Branch if Equal
BIT		24		2C								logical BIT
BMI											30	Branch if Minus
BNE											D0	Branch if NOT Equal
BPL											10	Branch if Plus
BRK									00			BReak
BVC											50	Branch if oVerflow Clear
BVS											70	Branch if oVerflow Set
CLC										18		CLear Carry
CLD									D8			CLear Decimal
CLI									58			CLear Interrupt
CLV									B8			CLear oVerflow
CMP	C9	C5	D5	CD	DD	D9	C1	D1				CoMPare
CPX	E0	E4		EC								ComPare X
CPY	C0	C4		CC								ComPare Y
DEC		C6	D6	CE	DE							DECrement
DEX									CA			DECrement X
DEY									88			DECrement Y
EOR	49	45	55	4D	5D	59	41	51				logical Exclusive OR
INC		E6	F6	EE	FE							INCrement
INX									E8			INCrement X
INY									C8			INCrement Y
JMP				4C								JuMP (Indirect 6C)
JSR				20								Jump to Sub Routine
LDA	A9	A5	B5	AD	BD	B9	A1	B1				LoaD Accumulator
LDX	A2	A6		AE	BE							LoaD X
LDY	A0	A4	B4	AC	BC							LoaD Y
LSR		46	56	4E	5E							Logical Shift Right
NOP									EA			No OPeration
ORA	09	05	15	0D	1D	19	01	11				logical OR Accumulator
PHA									48			PusH Accumulator
PHP									08			PusH Processor status word
PLA									68			PuLI Accumulator
PLP									28			PuLI Processor status word
ROL		26	36	2E	3E					2A		ROtate Left
ROR		66	76	6E	7E					6A		ROtate Right
RTI									40			ReTurn from Interrupt
RTS									60			ReTurn from Subroutine
SBC	E9	E5	F5	ED	FD	F9	E1	F1				SuBtract with Carry
SEC									38			SEt Carry
SED									F8			SEt Decimal
SEI									78			SEt Interrupt
STA		85	95	8D	9D	99	81	91				STore Accumulator
STX		86		8E								STore X
STY		84	94	8C								STore Y
TAX									AA			Transfer Accumulator to X
TAY									A8			Transfer Accumulator to Y
TSX									BA			Transfer Stack pointer to X
TXA									8A			Transfer X to Accumulator
TXS									9A			Transfer X to Stack pointer
TYA									98			TRansfer Y to Accumulator

Fig. 5

```

2000 INPUT "START ADDRESS IN HEX"; N$
2010 IF LEFT$(N$) "#" THEN N$ = "#" + N$
2020 N=VAL(N$)
2030 CLS:?:?" LOCATION CONTENTS NEW CONTENTS"
2035 DOKE#26D,#BBF8:POKE623,24
2037 PRINT:PRINT
2040 REPEAT
2050 PRINT HEX$(N) " "HEX$(PEEK(N))" ";
2060 INPUT A$
2065 IFA$="EEE"THEN20120
2070 A$="*" + A$:A=VAL(A$)
2090 IFA 255 THEN PRINT" ONE BYTE AT A TIME!":GOTO20050
2100 POKEN,A
2110 N=N+
2120 UNTILA$="EEE" OR N #B3FE
2130 DOKE#26D,#BB80:POKE 623,27

```

BASIC Machine Code Loader

Telephone Directory

By M. Caldwell



This program allows you to enter names and numbers into an array. It will fit into any model Oric but I don't know if when the directory is full if it will fit into the 16K Oric.

The program is menu driven. Here is a list of the options:

- (1) start a new directory. This clears the memory and allows a new directory to be inputed.
- (2) to view the directory. This displays the directory in alphabetical order.
- (3 & 7) save on tape. This will save the directory on to tape, no. 7 is at 300 baud.
- (4) to search for a name. With this you can enter part of a name or the full name.
- (5) to add more names. Allows you to add to the directory.
- (6) STOP.

Note that options 2 and 4 must only be used if there is something in the directory.

When reloading the program do not use RUN, instead use GOTO 400, using this will not clear the array.

Routines
100-300 new directory
300-400 Sorts directory in to alphabetical order
400-600 The menu
1000-2000 Searches for a name
2000-2500 adds more names to directory.

From now on a number will not be forgotten.

Perhaps with a bit of thought this program could be changed to make, for example, a small database or perhaps a diary. I will

leave you to think about that as I must go and phone my friend, . . . I wish I had put his number in my directory, what was it I wonder?



Listing on page 31

Getting the Most from Oricbase

By Geoff Phillips

Over the last few months, I have received a number of enquiries about using the Oricbase database program.

Although the manual has been extended, Oricbase can still be difficult to understand, so this article will explain some of the features of the language.

Syntax of commands

Like all computer languages, Oricbase requires that the sentences you type be formatted in an exact way. One very common mistake is to leave out spaces between parts of an Oricbase sentence. Page five of the new manual does not help, since it shows the example 'FIND AGE <31' which should be 'FIND AGE < 31'.

Quotemarks (") should be put around any item that includes spaces, for example an address, when typing an Oricbase sentence. For instance:

```
FIND NAME = "FRED BLOGGS".
```

How to use FIND

Used on its own, FIND looks through your file of information, marking the records found and

determining how many fit your FIND criteria.

After doing a FIND these records will remain marked, so that further Oricbase commands can make use of the selection made.

AND allows you to find records in your file depending on two (or more) things being true about that data. AND can either be used with FIND, or as a later supplement to further restrict the records found. For example: FIND NAME > "RONAYNE" AND AGE > 45.

OR on the other hand allows two (or more) different conditions to affect the records found. If, for example, you want either people who are less than 21 or who have a rating of 3, then type:

```
FIND AGE < 21 OR RATING = 3.
```

It must be realised that FIND has a rather limited format, as it compares a field name to an actual value. FIND cannot use registers or compare between different fields. The following example is INVALID: FIND AGE > MAXAGE.

BEGIN, ATRECORD and END

These three commands are the source of much misunderstanding.

The revised Oricbase manual gives an incorrect flowchart as to the use of BEGIN . . . END, as it should show:

- (1) BEGIN occurring once.
- (2) ATRECORD being done for each record previously found by a FIND command.
- (3) END occurring once.

The important one to look at is 'ATRECORD', or just 'ATR' which is followed by a variety of commands to be obeyed once for every record that FIND has selected. When simply updating the file, then MOVE is used, moving an actual value to one of the fieldnames.

In other words, if we want to correct an error on a record, such as a new telephone number in a particular record, we must:

CAPTAIN TANEX & ORIC-Ω

THERE'S NOT ENOUGH TIME TO MAKE ANOTHER SEARCH THROUGH THE CATACOMBS

I SUPPOSE HAVING WON THEM I SHOULD PUT THEM ON

THESE BOOTS MUST BE THE "TALENTS" I'M SEARCHING FOR

I HAVE YOU NOW

THE EYE OF TANGER

CAPTAIN TANEX!

SWISH

WHAT!

I'VE GOT TO GET AWAY

BUT IT COVERS THE EXIT

AND IT MOVES SO FAST

MOLTEN SILK FROM ONE OF ISSDPOL'S SPINNERETS CATCHES OUR HERO'S NEW BOOTS

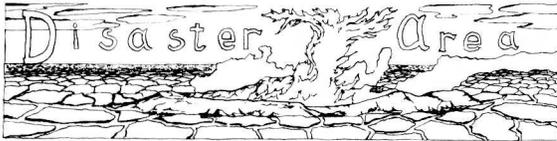
LEAVING HIM A MERE UNTALENTED MAN AGAIN

MEANWHILE..

IN THE TUNNEL SYSTEM..

ORIC SEEMS TO BE TAKING THINGS EASY

Regulars



The following are corrections for programs in issue 4.

Torpedo Attack

Line 780 should be PLOT X,C,124.

Lines 3500 onwards: as far as I can see X will never be false so its presence in the conditionals is superfluous.

3D Bar Chart

Line 50: Colon missing between print and input statements.

Line 630: a comma appears from nowhere.

Line 5060: there's a square bracket instead of a round one.

Line 5100: there's a comma instead of a colon.

There are X's for *'s throughout the program and S's for \$'s in places.

Agrenon

The line that is numbered 31002 should be 31030. As it is it causes an Illegal Quantity Error in your moment of triumph.

Character Generator Program

The semi-colons on lines 340 and 350 want losing.

Figures

Line 150 should read: 150 DRAW X1-X+5,Y1-Y+5,1 I think.

"Frog-Run" and "Word-1" had the following printing errors in them:

1/8 should be <

3/8 should be >

1/4 should be ↑

What are Softbacks?

Softbacks are NEW programs for your computer.

For running a home, club or small business . . .

Account Book - The money management program. Budgeting and record keeping by automatic double entry. Produces reports on screen or printer. Easy to use and menu driven throughout.

And for the younger user . . .

Story Book - A challenging game that develops reading skills. Suitable for any level from early reader onwards. You choose the text, the game is to reconstruct it by guessing the words.

Picture Book - A colourful spelling game for young children. Build up pictures by typing words and seeing the objects appear "by magic".

Softbacks replace books!

To: SOFTBACKS, PO Box 257, Watford, WD1 3LQ. 

Please send: ACCOUNT BOOK £9.95* []
 STORY BOOK £4.50* []
 PICTURE BOOK £4.50* []
 Further details (I enclose sae) []

DEDUCT 50p from total for more than one program.
I enclose cheque payable to Softbacks for £.....

My ORIC is 16K [] 48K []

NB. Account Book is for 48K only.

Name

Address

.....

.....

* INTRODUCTORY PRICE applies until 30th April. 274

Parallel to RS232C interface

By P. J. W. Cooper

One shortcoming of the Oric is that it has no serial printer interface. This is not usually a problem unless a serial printer is the only machine available. This circuit was designed to convert the signals available at the Oric printer socket into signals suitable to drive an RS232C serial printer.

The term RS232C defines a serial interface system that has a range of uses, including modems, printers and V.D.U's. Interconnections are made via a 25 way 'D' type connector with the socket at the computer end. The most commonly used pins are shown in Fig 1, together with their abbreviations.

The data sent to the printer on Pin 3 will be in serial form. The order in which the code is sent is illustrated in Fig 2. The first bit transmitted is always the start bit. This is followed by seven data bits (least significant bit first), and the parity bit. The parity may be odd or even depending upon the standard in use, and its purpose is to enable simple transmission errors to be detected, by making the total number of logic '1's odd or even. Finally, 1, 1½ or 2 stop bits are transmitted, which leaves the line ready to pass the next char-

acter code, beginning with it's start bit.

Handshaking is utilised via Pin 4 on the connector, entitled request to send (RTS). This allows transmission when set at logic high (1), and is the only form of handshaking provided for in this design.

Design

The interface design is based on an Intersil universal asynchronous receiver/transmitter (UART) integrated circuit. This one chip does all the work of coding data into serial form and vice versa. The only extra circuitry required are a clock oscillator, RS232 level converters and an acknowledge (ACK) pulse generator. The interface circuit, which uses only half of the UART chip, is shown in Fig 4.

Full use of the UART chip could be used to implement a two-way communication link (via a modem?), but this would entail incorporating the UART into the memory map of the Oric, and some extra software would be needed.

Operation

The operation of the printer interface is briefly as follows

When data is required to be sent, the printer strobe line is pulsed low. The character code is then latched off the parallel printer data lines, and stored in the transmit buffer register of the Uart. It is then automatically sent, along with start, parity and stop bits, from Pin 22. As the serial transmission rate is likely to be much slower than the normal workings of the computer, the generation of the acknowledge pulse is delayed until the transmit buffer register is empty (TBRE goes high). The two operational amplifiers do the necessary level conversion.

The Baud rate is selected from the various rates available by connecting the appropriate output of the 12-stage counter to the clock input of the Uart. The clock frequency runs at 16 times the Baud rate.

The transmission standard used by the Uart is determined by the insertion of various straps, Fig 3 shows the combinations available.

Features

Fig 1 RS232 Connector (Relative to Printer)

Pin Number	Abbr.	Description	Signal Direction
1		Protective GRND	
2	TD	Transmitted data	O/P
3	RD	Received data	I/P
4	RTS	Request to send	O/P
5	CTS	Clear to send	I/P
6	DSR	Data set ready	I/P
7		Signal GRND	
8	CD	Carrier detected	I/P

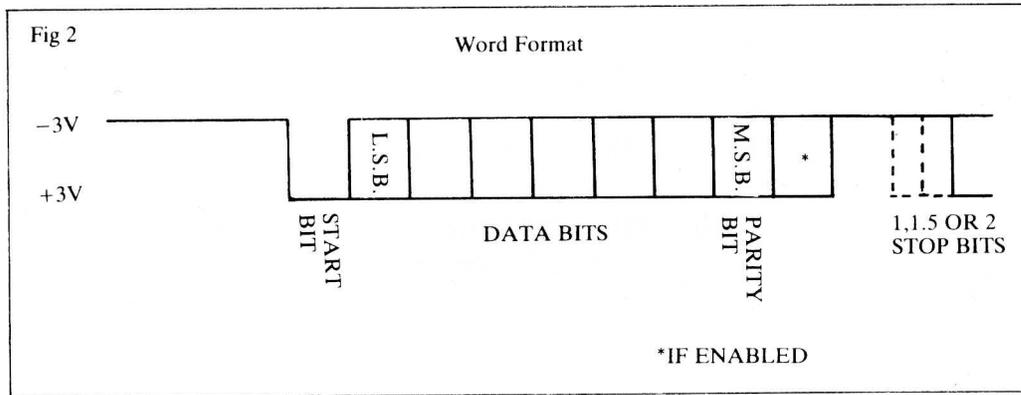


Fig 3 Transmission standard selection

	Character length	5	6	7	8	Bits
CLS1		L	H	L	H	
CLS2		L	L	H	H	

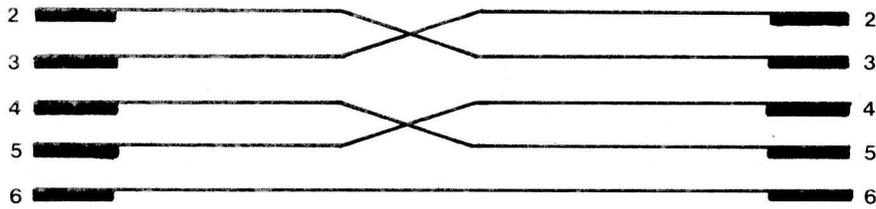
SBS I/P	Character length selected		STOP BITS
	5 BITS	6,7 OR 8 BITS	
L	1	1	}
H	1½	2	

Parity Selection		
PI	EPE	Effect
L	L	Odd Parity
L	H	Even Parity
H	L	Parity Inhibited
H	H	Parity Inhibited

Features

Fig 4 See large circuit diagram

Fig 5 Connecting Cable
(Interface to Printer)



Pin Definitions (IC3) Table 1

Vcc	+V Supply
N/C	No connection
GND	Ground
RRD	Receive register disable
RBR 1-8	Receive buffer register outputs
PE	Parity error
FE	Framing error
OE	Overrun error
SFD	Status flag disable
RRC	Receive register clock
DRR	Data received reset
DR	Data received
RRI	Receive register input
MR	Master reset
TBRE	Transmit buffer register empty
TBRL	Transmit buffer register load
TRE	Transmit register empty
TRO	Transmit register output
TBR 1-8	Transmit buffer register inputs
CRL	Control register load
PI	Parity inhibit
SBS	Stop bit select
CLS 1-2	Character length select
EPE	Even parity enable
TRC	Transmit register clock

Features

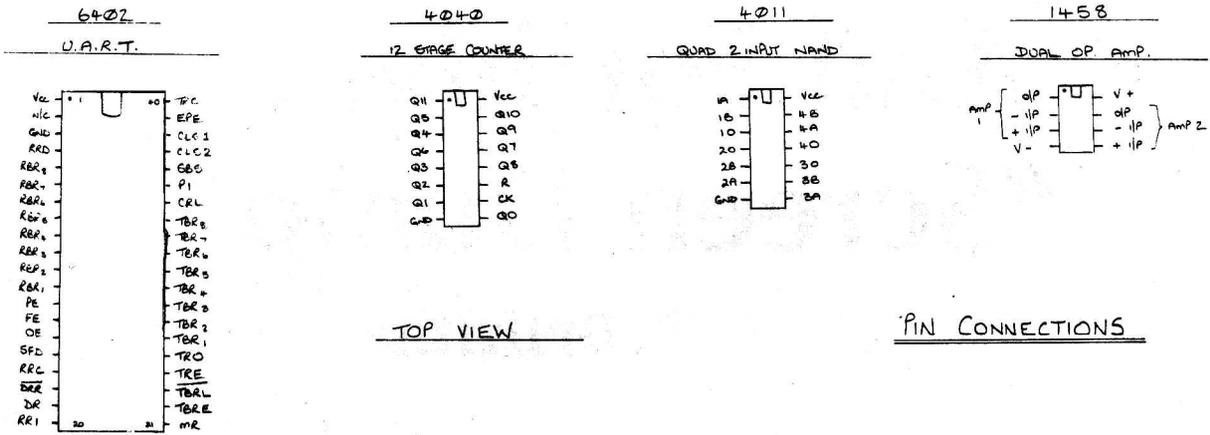
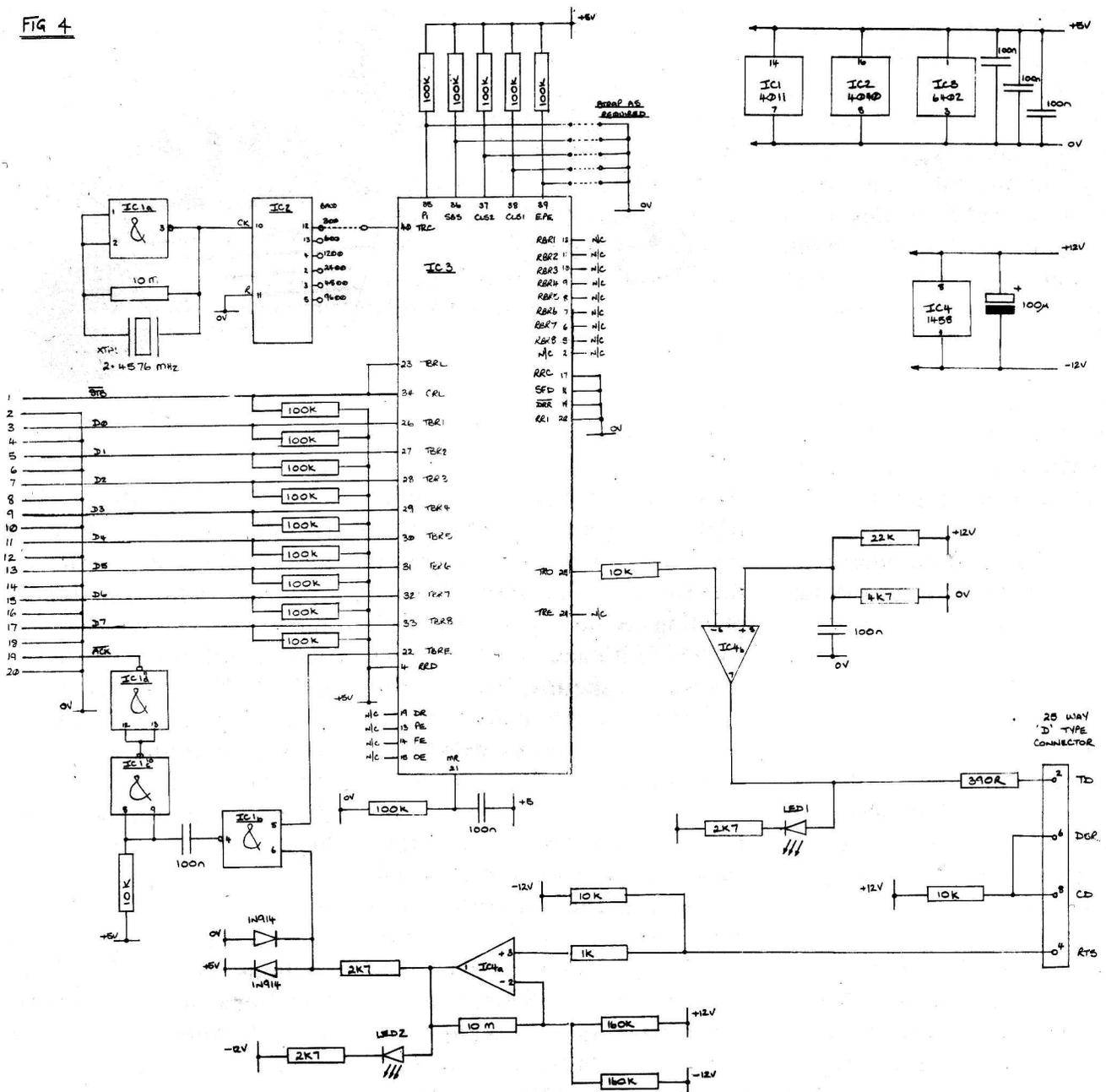


FIG 4



And Forthly. . . "Screen Dump"

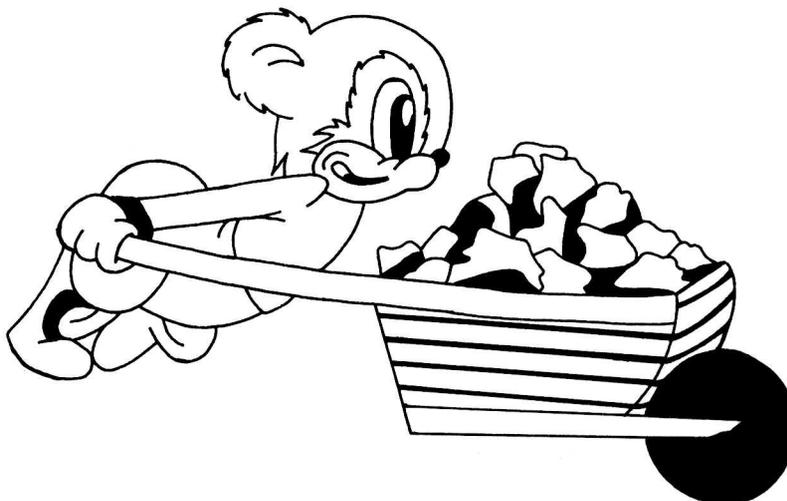
By K. W. Griffiths

When using a computer in either an immediate mode for one-off calculations, or programming mode, there is nothing more frustrating than finding you have a screenful of data, answers, results, etc. and have to resort to writing out the information. A much more useful, professional and accurate method is to dump the contents of the screen direct to the printer. Most computers and languages lack this facility and Forth is no exception. This program overcomes that limitation by allowing the Text screen to be compiled by the printer in immediate or program mode.

For those as yet unfamiliar with Forth I will detail the workings of the program.

In order for the definition to work, screen one of the option-screens supplied with the ORIC-Forth package should first be loaded and compiled. A reminder to this effect can be seen in line one when the program is listed.

Lines 2 and 3 contain two machine code words called "KEYON" and "KEYOFF". These overcome the tiny bug in the ORIC, which causes an odd



character to occasionally appear from the printer, by switching off the Keyboard-scanning routine (the cause of the bug) when printing is in operation, and switching it back on when printing has finished. Of course this problem doesn't occur on the Atmos and so these words can be ignored.

Lines 5 and 6 set up four important variables which will be used in the definition; they are:

Screenstart initially set to 0
Linestart initially set to 2
Line-end initially set to 38
Screen-end initially set to 49119

Line 7 starts the definition. DUMP is the title of the definition. PON switches the printer on, the definition for which appears on option screen one mentioned earlier, and KEYOFF switches the keyboard-scan off. SCREEN-START is initially set to 48040 (the start of the Text-screen location) and a BEGIN-UNTIL loop is then started.

The screen of course is 40 columns wide, the first two however contain information as to background and foreground colours. This program prints only what is in the next 38

Software

columns in each line, i.e. it ignores the first two.

After the word BEGIN, a character space is output to the printer (the reason for this is that the ORIC printer is normally a 40 column printer and with information appearing only on the last 38 columns, the hard-copy would look slightly out of balance, therefore I have included one space before the printing of data starts and one space after it has finished on each line to balance the print-out).

The variable SCREENSTART is now placed on the stack (48040) together with LINE-START (2) and LINE-END (38). All three are added together leaving a value of 48080 on the first line of the screen. SCREENSTART (48040) and LINE-START (2) are again placed on the stack and added together leaving (48042), the first column of the first line of the screen, which we wish to print. We now have two values on the stack in the correct order for a DO-LOOP. After the word DO the word I appears which places the loop index on the stack. C@ fetches the byte from that address and places it on the stack. DUP duplicates this item so that we have two copies of the value. 32 is then placed on the stack. < is used as a comparison to see if the second value down on the stack is less than that of the top one. If it is it implies that the value at the appropriate screen address is not a printable character but a control character (i.e. change of foreground col-

our, etc.) and the program enters an IF-ELSE-ENDIF structure and prints a blank space on the paper. If it is not less than the top value it progresses to the ELSE part of the structure and merely takes the value now left on the stack and prints it using EMIT. (i.e. if it finds the value 65 it will print the letter A). We then reach the word LOOP which now returns to the word just after DO to continue printing the rest of the line unless it has reached the end of the line in which case it prints the second blank space to help with formatting on paper as mentioned earlier. SCREENSTART, LINESTART and LINE-END are all added together as previously but this time the value is stored back into SCREENSTART thus giving a new start position (i.e. the next line down). This value is again placed on the stack and 1 is subtracted from it. SCREENEND is placed on the stack and we now have two values: the bottom one being the start of the next line -1 (i.e. the end location of the previous line) and the top one being 49119, the end location in memory of the screen. They are compared using the word =. If they are not equal the procedure is repeated to print-out the next line. If they are equal printing has finished. A carriage return then takes place with 13 EMIT (13 being the control character for a carriage return), KEYON switches the keyboard scan back on and POFF turns the printer off.

To use the program in immed-

iate mode simply type DUMP and whatever is on the screen will immediately be printed out on paper. The program can also be used in your own programs either by including the word DUMP on its own or by including it in an IF-ELSE-ENDIF structure which you might like to print a prompt on screen such as "HARD-COPY Y/N?" and wait for your response i.e. yes or no.

The routine was written for the "ORIC-1 MCP-40 4 colour printer" and any other standard printer.

Important Note to all Oric-Forth users

We are constantly searching for Forth programs and articles, particularly tutorial ones, to print in the magazine. So if you have written something using Forth, even if it is just a small utility or hint we would be delighted to hear from you.

A number of people have asked us where they can get further information and help using Forth. I would strongly recommend the Forth Interest Group (F.I.G. UK) who can be contacted at 24 Western Avenue, Woodley, Reading, RG5 3BH. Annual subscription is £7 and is well worth it.

Features

And Forthly . . . Listing

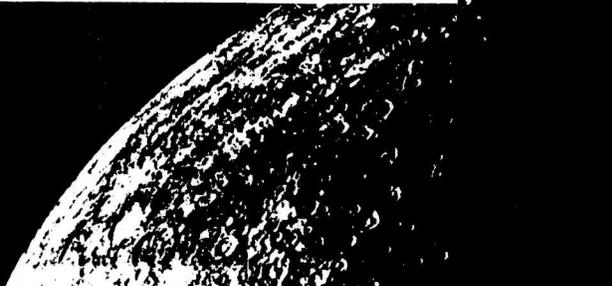
SCREEN DUMP

```

SCREEN1
0 ( Text Screen To Printer Dump )
1 ( Ensure OPTION SCREEN ONE is loaded
  before this one )
2 HEX CREATE KEYOFF B586 , CA20 , A6E6
 , 4CB5 , 0444 , SMUDGE
3 CREATE KEYON B586 , 0420 , A6E8 , 4C
 B5 , 0444 , SMUDGE
4 ( KEYOFF = Keyboard scan off. KEYON
 = Keyboard Scan on. )
5 DECIMAL 0 VARIABLE SCREENSTART 2 VAR
 IABLE LINESTART
6 38 VARIABLE LINE-END 49119 VARIABLE
 SCREENEND
7 : DUMP PON KEYOFF 48040 SCREENSTART
8 BEGIN 1 SPACE SCREENSTART
9 @ LINESTART @
10 LINE-END @ + SCREENSTART @ LINESTA
 RT @ +
10 DO I @ DUP 32 < IF 32 EMIT ELSE EMIT
 T ENDIF LOOP 1 SPACE
11
12 SCREENSTART @ LINESTART @ LINE-END @
 + SCREENSTART @
13 SCREENSTART @ 1 -- SCREENEND @ = UNTI
 L 13 EMIT
14 KEYON POFF ;
15 ;S
    
```

ORIC-1 FIG-FORTH

STARSHIP



- * Fast-action classic arcade game
- * 100% Machine-code
- * 8 different alien squadrons
- * Hi-score table
- * Full colour high resolution graphics

You are the commander of 3 heavily-armed starships defending your sector from waves of attacking alien spacecraft from the war-fleet of the Cirellian Empire. When all 8 squadrons have been destroyed, another wing moves in to the attack.

COMPOSER 48K	£6.50
STARSHIP 48K	£6.50
GAMESPACK 48K (7 games)	£7.00
GAMESPACK 16K (5 games)	£5.95

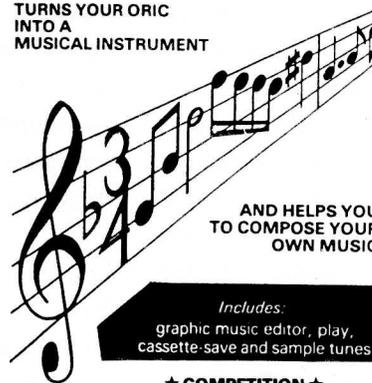
Send cheque or P.O. to SECTOR 7 SOFTWARE, P.O. BOX 8, NEWTON ABBOT, DEVON.
Trade enquiries welcome Tel: 06267-4534.

SECTOR 7 SOFTWARE

Composer

A VERSATILE MUSIC PROCESSOR
for the ORIC 48K

Turns your ORIC
into a
musical instrument



AND HELPS YOU
TO COMPOSE YOUR
OWN MUSIC

Includes:
graphic music editor, play,
cassette-save and sample tunes.

★ COMPETITION ★

ORIC MUSIC COMPETITION

First Prize £100.00
Second Prize £50.00.

Will be awarded for the most entertaining piece of original or arranged music, produced using COMPOSER.

Details on cassette insert.

3 part harmony, any key, 3/4 or 4/4, up to 80 bars, full control over sound and speed.

New Products Review

Oric Atmos



Just as we were going to press, I managed to get hold of Oric's latest offering – the Atmos. This has been such a well kept secret that half of Oric's own staff knew nothing about it and even I was surprised by its release. Although rumours have been floating about for some time concerning the new Oric ROM, I did not expect them to launch a new machine to go with it.

When I say 'new machine', I should really say 'new model' as

what Oric have done is to listen to what the critics have been saying about the existing machine and to apply this information in a constructive way.

The first thing that you notice is the keyboard, which has been upgraded to a full moving key mechanism. This is a great improvement on the old keyboard, (which itself was far superior to the Sinclair rubber membrane) and means that the machine can be considered for serious word-processing (how

about an 80 column screen next, please!) and even more significantly, at a stroke, wipes out any competition from the Commodore 64 and Acorn Electron on that count. The layout is identical to the original model except for the addition of an extra key labelled "Function". If pressed, in command mode or in a program, the key is ignored but clever programmers can access it through machine code by reading the keyboard matrix directly. This would allow it to be used for any desired function

News

by the programmer and perhaps later Oric Software would make use of it.

Now we come to the most exciting part of the machine – the new ROM. This announces itself on power-up as version V1.1 and is a souped up version of the standard V1.0 ROM. The first thing you notice is that screen handling is about 30% faster which means that if you listed a program on the old ROM, it would glide in a stately fashion up the screen, on the new ROM it scrolls just too fast to be readable and so you have to keep pressing space to freeze the listing. I also noticed that cursor always appears two positions in on the screen instead of at the far left. This saves the cursor have to skip over the first two protected positions every time a new line is begun.

All the original commands are still present and a couple more have been added. More importantly, virtually all of the bugs have been corrected. This means that TAB now works properly and the printer no longer prints spurious characters. More obscure problems such as with ELSE and VAL/STR\$ have been cured which allow you to program with much more confidence.

Most of the changes and improvements will be found in the cassette systems. Firstly, there are two more letters which can be added to the CLOAD command. V will verify a specified program on tape with one

already in memory. Any differences will be counted and printed at the end. J will join a Basic program on tape with one already in memory although you do have to be careful with line numbers. Whether you are Loading, Verifying or Joining, if a file is found on the tape which is not the one you are looking for, it's name is flashed up on the top of the screen instead of just being ignored. If there are any loading errors, you are told at the end and the system will try to re-link a faulty program rather than just print up a screen full of U's.

Two commands have been added to handle cassette data. These are STORE and RECALL which allow you to dump and load from arrays. For instance, if you have an array called A\$, you could save it on tape with: STORE A\$, "FRED",S which means store the array A\$ on tape with the filename FRED at slow speed. Conversely to read it back you would enter: RECALL B\$, "FRED",S which means read the file FRED from tape into the previously dimensioned array B\$ at slow speed. If the array B\$ has not been created, you will get an error message. Although these commands, to my mind, are not as useful as full record handling commands, they are a vast improvement on what was previously available, i.e. Nothing!

There is a slight problem reading tapes created on old Orics as the

new ones expect slightly more information in the tape header and sometimes give an error if they don't get it. One useful improvement is that you can load and save from within a program without dropping out of it at the end. Also if you save a Hi-Res screen, you don't get the 'SAVING' message scrawled across it in graphics.

Other improvements include PRINT @ (pronounced 'Print at') which allows you to print at any required X,Y co-ordinate on the screen, and POS which will now tell you the position of the cursor on the screen or the print head of your printer. Virtually every bug I could think of has been corrected including POKE which did not like Hex numbers and FILL which did not update the cursor position properly.

So, all-in-all, the Atmos is everything the Oric owner has dreamed of for the last year. The only thing I am not certain about is Oric's policy concerning existing owners who wish to upgrade their ROM's to the new version. It should be noted that many machine code programs written on the old machine will not function properly on the new machine due to changes in zero-page usage and ROM locations. It may be wise to check before buying which version machine a tape has been designed for.

**By
Paul B. Kaufman**

ORIC Software

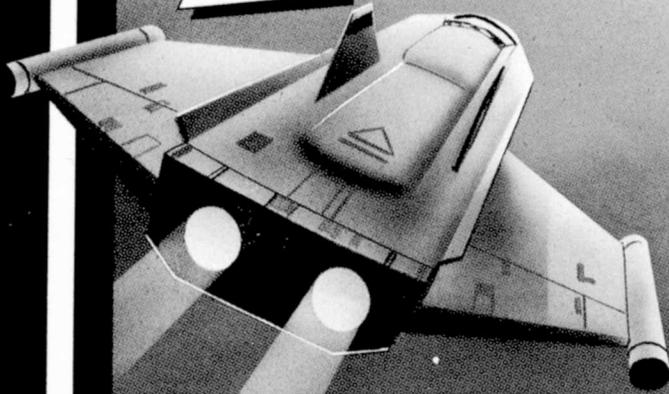
TANSOFT

ultima ZONE

A 100% machine
space trilogy
Walkons, avoid the
bouncing Brunescos
your way through the
satellite zone. Requires
48k Oric.

£8.50

inc. V.A.T. post free



THE HOBBIT

In co-operation
with
Melbourne House.

At last, this best selling
adventure is available for
the Oric based on J. R. R.
Tolkien's book 'The
Hobbit' (included) a
complete text and
graphics adventure.
Requires 48k Oric.

£14.95

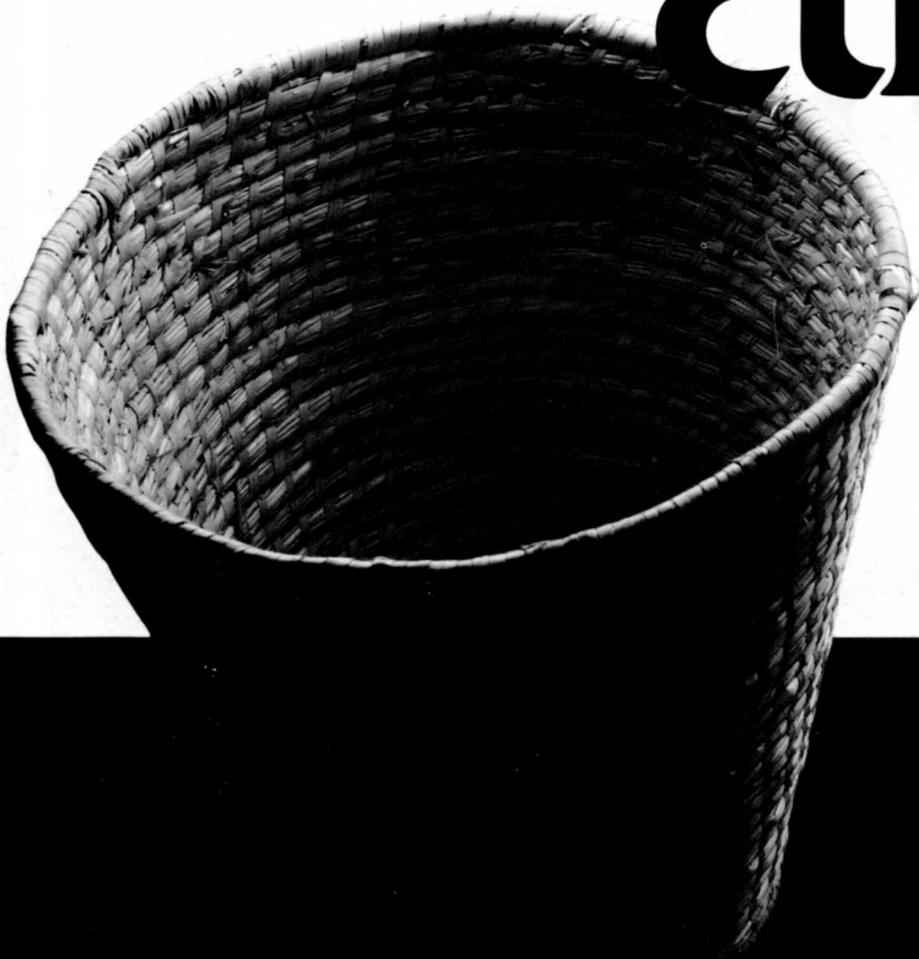


Our software is available from all Oric dealers and
most good software suppliers. In case of difficulty
please contact us on Teversham (02205) 2261 or write to us at:

TANSOFT

Unit 1 & 2, Techno Park, Newmarket Road, Cambridge

Three c amm



A rubbish bin for your old
has been.

Yet more money
bolt

■ Believe it or not, computers often suffer from amnesia.

99% start off with large enough memories, but operating functions like text, colour, sound and more particularly high resolution graphics, take large bytes out of them. Leaving very little "useable" memory for programming and games.

Not so the new Oric Atmos 48K.

This is the one home computer that takes these normal working functions in its stride.

Unlike other home computers it uses the highly sophisticated serial attribute handling method used by Viewdata and Teletext,

in which the attributes are stored on the screen alongside the data, instead of taking up space in the memory.

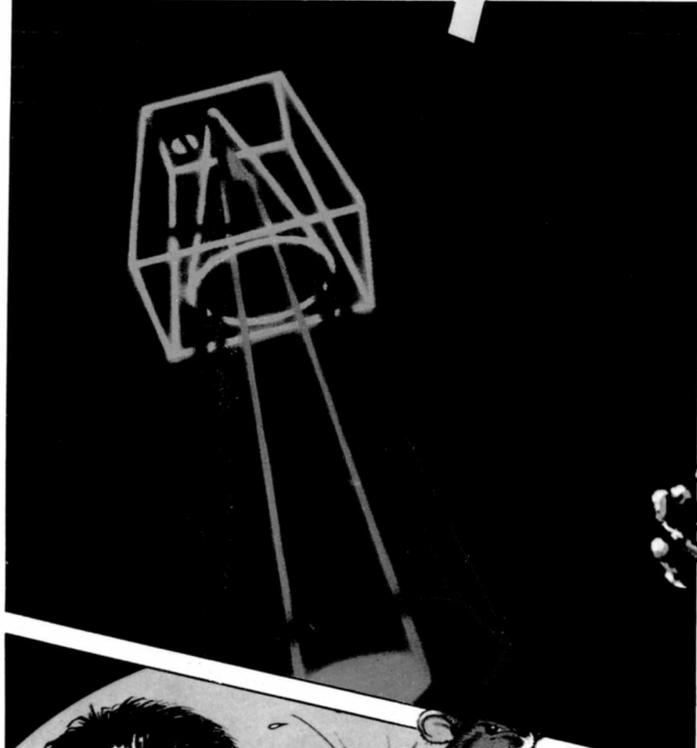
Thus the Oric Atmos never offers less than a healthy 37K of useable memory – even when the new colour printer and disc drive unit are attached. (Technical buffs see details overleaf).

So it rivals the performance of the supposedly larger, more expensive Commodore 64K, which unfortunately loses 26K of its "elephantine" memory in high resolution graphics.

It beats its immediate competitors like the Sinclair Spectrum, Dragon 32K, Vic 20 and Atari 600.

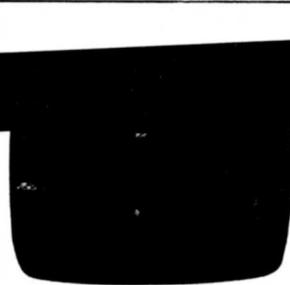
ORIC

SOFTWARE

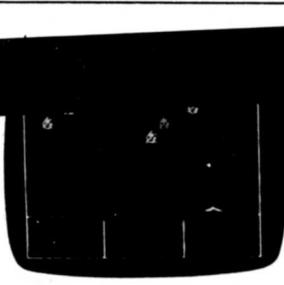


TANSOFT

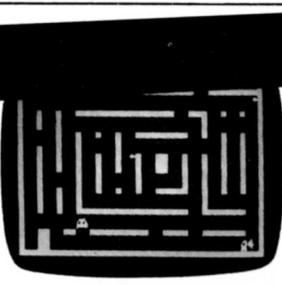
OricSoft



DEFENCE FORCE Pilot your space-craft through hostile alien territory, fight off the enemy coming at you from both sides. **£7.95 inc. VAT—48K.**



ULTIMA ZONE A 100% machine code space trilogy—shoot the Walkons, avoid the bouncing Brunes, battle your way through the satellite zone. **£8.50 inc. VAT—48K.**



ORICMUNCH Munch the bugmonsters and eat the power pills, clear the maze to reach the high score. **£7.95 inc. VAT—16K or 48K.**



SUPER-ADVANCED BREAKOUT An excellent version of this popular arcade game. Features normal and progressive modes and selectable skill levels. **£4.50 inc. VAT—16K or 48K.**



RAT-SPLAT A truly revolting game! Chase the rats down the sewers, splat them with your hammer, spray the monsters with your aerosol can, and rescue the cheese! **£7.95 inc. VAT—48K.**



Business



ORICBASE A complete database system—use it as an electronic filing cabinet. A sophisticated query language allows you to search and sort your data, keep running totals and print reports. Includes manual. **£14.50 inc. VAT—48K.**



ORIC-CALC A full spread sheet program allows you to enter data sum columns or rows and apply complex formulae. Change one item and watch how all your other figures are affected. Includes manual. **£14.50 inc. VAT—48K.**



AUTHOR A powerful word-processor program which includes word-wrap and word count. Ideal for reports or letters. Full control over tabs, formatting etc. Includes manual. **£14.50 inc. VAT—48K.**



Adventure Games

THE HOBBIT At last, this best selling adventure is available for the Oric. Based on J.R.R. Tolkien's book 'The Hobbit' (included). A complete text and graphics adventure. **£14.95 inc. VAT—48K.**



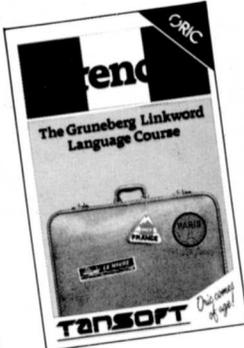
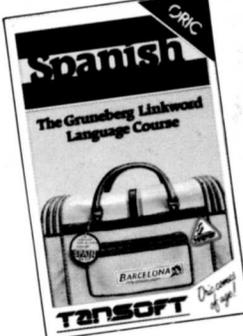
HOUSE OF DEATH An impressive follow-up to Zodiac, discover the secrets of the haunted house, avoid the witch and the axe murderer and come away with incredible treasures. **£9.99 inc. VAT—48K.**



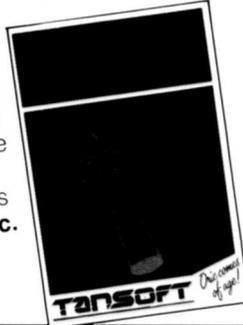
SOFTWARE

General Interest

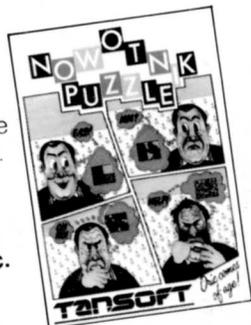
TANSOFT LANGUAGE MASTERCLASS If you are going abroad for pleasure or business these courses will give you a basic understanding of the language. Based on the new Gruneberg Link-Word method.
 Spanish, Italian, German, French, supplied with program cassette and spoken vocabulary pronunciation. **£12.95 each inc. VAT—48K.**



ORIC-CAD Computer Aided Design for your Oric. Display an object on the screen, enlarge, rotate it or shade it in. Will work with Oric printer. Includes full instructions. **£9.99 inc. VAT—48K.**



NOWOTNIK PUZZLE The Rubik cube of the screen. Can you move the coloured blocks to solve the puzzle? Hours of frustrating fun. **£6.90 inc. VAT—48K.**

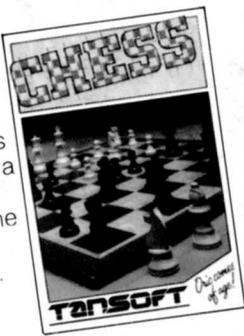


Other Games

MULTIGAMES PACK 2 Five further games for those with a few minutes to spare: Hangman, Lunar Lander, Substrike, Racer and 3D-Link—4. **£6.90 inc. VAT—48K.**



CHESS Superb graphics makes this game a pleasure to play. Five skill levels for those who want a quick game and those who have time for a more challenging game. **£9.99 inc. VAT—48K.**



ZODIAC Discover the twelve cunningly disguised signs of the zodiac which will lead you to six treasures. Full puzzles, puns and misleading clues. **£9.99 inc. VAT—48K.**



FORTH A full implementation of this exciting new computer language. Includes editor and assembler plus language extensions. Includes manual. **£15.00 inc. VAT—48K.**
 Please state whether for use with Oric 1 or Atmos.



ORIC-MON For the enthusiast who wishes to get to grips with machine code. Includes mnemonic assembler/disassembler, block move and verify, data and text entry and full machine code



cassette and printer handling. Includes manual. **£15.00 inc. VAT—16K or 48K**

Hardware

The new Oric Atmos re-affirms Tansoft's heart-felt belief that the only justification for the support of one home Computer is that Oric has the recipe for unrivalled success.

Furthermore, with the recent introduction of the 4-colour Printer and Micro Disc Drive, the Atmos now gives you full system capabilities.

Pick up the Oric hardware brochure and see why Tansoft is totally devoted to you!

Prices

Oric Atmos 16K	T.B.A.
Oric Atmos 48K	£170.00
Oric 4-colour Printer	£150.00
Oric Micro Disc (with interface)	£260.00
Printer Pens (set of 4 colours)	£3.99
Printer Rolls	£1.99 each

The Oric Owner bi-monthly magazine is the most important publication for enthusiasts of the Oric. It has the manufacturer's blessing because it caters for every interest. So you can expect to find the latest developments and new releases, 'behind the scenes' interviews with the V.I.P.'s, answers to those irritating technical problems, program suggestions and much, much more!



One issue is free with your Oric. The other issues may be purchased from your Oric stockist. A year's subscription of 6 issues is available from Tansoft at only £10.00 (£15.00 overseas). Can you imagine life without it?

All the products in this catalogue are available from your nearest Oric stockist. In case of difficulty, you may order directly from Tansoft at the address shown below.

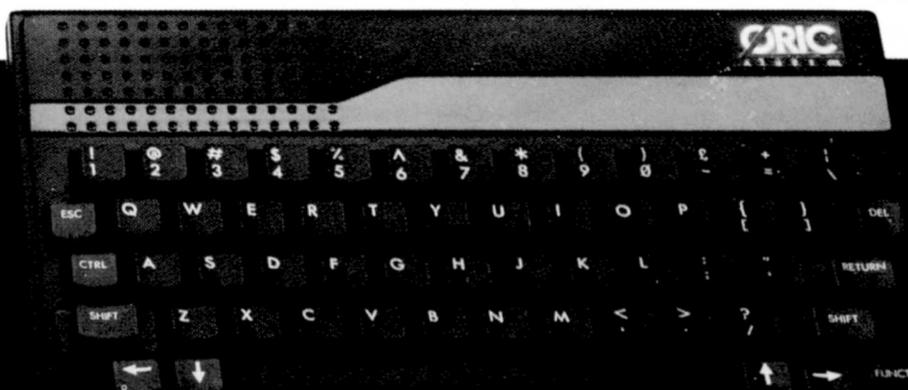


Prices and data correct at time of going to press—February 1984. Tansoft reserves the right to alter prices and specifications and to withdraw products without notice.

TANSOFT

Tansoft Ltd.,
Units 1 & 2, Cambridge Techno-Park,
Newmarket Road, Cambridge CB5 8PB

ures for esia.



for yet more
ons.

The new Oric Atmos 48K.

It beats the Lynx 48K, which costs over a third more, yet loses 34K.

It even beats the Acorn Electron and the BBC Micro which costs more than twice as much, yet loses 23K in high resolution graphics.

And while this may surprise you, it's totally in keeping with a company recognised in the computer industry for performance and innovation.

Like its predecessor, the Oric-1, the Oric Atmos has the powerful loudspeaker and amplifier unit that prompted "Which

Micro" (November issue) to comment... "Its sound facilities have more in common with those of the £400 Beeb, than the rather pathetic beep of the Spectrum. At full volume it can compete with most arcade games..."

Yet the Oric Atmos 48K costs a mere £170, including all the leads and adaptors you need to get it going.

So if you're buying a computer, remember our name. We could save you a fortune on bolt-ons... or wastepaper bins.

The new Oric Atmos 48K. **ORIC**

Now we've whetted your appetite, here's something to get your teeth into.

Printer Technical Specifications

Printer/Plotting system	Ball Point Pen, 4 colour
Plotting speed: (horizontal)	52 mm/sec (2.05ips)
(vertical)	73 mm/sec (3.08ips)
Printer Speed	12 characters per second
Resolution	0.2 mm/step (0.00787 inch)
Effective plotting range	96 mm (3.804 inch) x axis, divided into 480 steps. (No limit in y direction)
Characters per line	80 or 40 text mode (determined by software in graphics mode)
Characters per line	INT (480/n+1) * 6 for 0=n=15
Accuracy (repetition)	0.2 mm max
(movement)	0.3 mm max
(distance)	0.5% max (x-axis) 1% (y-axis)
Pen life	250 metres (825 feet)
Parallel interface	8-bit parallel Uses STROBE and ACKNOWLEDGE
Temperature range storage	18.3 to 35°C (65 to 96°F) -40 to 71°C (-40 to 160°F)
Humidity range	10% to 80% relative non-condensing
Power supply	Switching power supply input 100-120 VAC 200-240 VAC
Dimensions	10 3/4" wide 6 7/8" deep 2 1/2" high

Atmos Technical Specifications

CPU	6502 A
Memory	Choice of 16K or 48K RAM
Memory (48K Model)	Minimum 48K RAM, max 64: 16K ROM external control signals allow use of full 64K RAM or maybe used externally to increase ROM/RAM
Language	Extended Microsoft basic
Keyboard	Typewriter style and pitch, 57 keys, standard computer layout, additional cursor control keys, autorepeat facility, tactile and acoustic feedback
Display	Output for B&W or colour TV, RGB output for colour monitor.
Text format	40 line x 28 rows
Character set	Similar to Teletext format, standard ASCII double height, flashing, 80 user definable characters
Graphics	240 x 200, 8 colours
Graphic Facilities	Points, lines, circles
Sound	Internal loudspeaker and amplifier. 3-Channel sound synthesiser envelope control, amplitude control 8 octaves, noise channel
Storage	Most cassette recorders via DIN socket 300 or 2400 BAUD. Disc Drive.
Interface	Centronics, expansion port, Hi-fi, RGB Monitor, UHF TV, cassette recorder
Other	Warm reset to regain control without clearing program or data

Micro Disc Technical Specifications

Formatted Capacity	160K bytes per side (double density as standard)
No. of Tracks	40 (80 available as option at a future date)
No. of Sectors	16
Bytes per Sector	256
Transfer Rate	250K Bits/Sec
Supports up to 599 files per side, four drives single or double sided, 40 or 80 track. User definable configuration allows mixing of drive types including 5 1/4" (five and a quarter inch) Extensive wild card facilities Copy allows merging of basic and machine code files	
<u>Utilities</u>	
The Utilities are as follows:	
1. Backup	Copy a whole disc
2. Copy	Copy a file to another
3. Del	Delete a file allowing wildcards
4. Dir	Display directory listing
5. Drv	Set the default drive number
6. Format	Format and initialise a disc
7. Load	Load a file (code data or basic)
8. Protect	Change protect status of file
9. Recall	Recall a basic array from a file
10. Ren	Rename a file
11. Save	Save a file (code, data or basic)
12. Store	Store a basic array as a data file
13. Sys	Change system configuration

Prices and data correct at time of going to press. Specifications on the above models may change without notice.

Available at Dixons, Laskys, Comet, Wigfalls, Rumbelows and all good computer stockists.



ORIC

Telephone Directory

```

470 PRINT:PRINT"5)TO ADD MORE NAMES"
480 PRINT:PRINT"6)TO STOP"
490 PRINT:PRINT"7)TO SAVE DIRECTORY ON TAPE AT 300B"
495 PRINT:PRINT"ENTER THE NUMBER NOW"
500 GETZ$:IFZ$<"1"ORZ$>"7"THEN500
510 IFZ$="1"THENCLEAR:GOTO100
520 IFZ$="2"THENGOTO300
530 IFZ$="3"THENCLS:PRINT"PRESS ANY KEY WHEN READY
TO SAVE":GETZ$:C SAVE"DIRECT."
540 IFZ$="7"THENCLS:PRINT"PRESS ANY KEY TO START":
GETZ$:CSAVE"D",S
550 IFZ$="6"THENCLS:STOP
560 IFZ$="4"THENGOTO1000
570 IFZ$="5"THENGOTO2000
590 END
1000 CLS
1010 PRINT:PRINT"YOU MUST ENTER THE FULL NAME OR JUST"
1020 PRINT:PRINT"PART OF IT EG SMITH COULD BE ENTERED"
1030 PRINT:PRINT"AS SM.INPUT NOW"
1040 INPUTL$
1050 M=LEN(L$)
1060 FORD=1TO300
1070 IFLEFT$(A$(D),M)=L$THENPRINTA$(D)
1090 NEXT:GOTO1110
1110 PRINT:PRINT"PRESS ANY KEY TO RETURN TO MENU":GETZ$
1120 GOTO400
2000 CLS
2010 FORD=1TO300
2020 IFLEFT$(A$(D),1)<CHR$(65)THEN2040
2030 NEXT
2040 FORZ=DTO300
2050 INPUT"NAME";B$
2060 INPUT"TELEPHONE NUMBER";C$
2070 CLS
2080 PRINTB$+" ";C$
2090 PRINT:PRINT"IS THIS CORRECT? Y\N"
2100 GETZ$
2110 IFZ$="N"THENGOTO2050
2120 A$(Z)=B$+" "+C$
2130 CLS
2140 PRINT"PRESS ANY KEY TO ENTER NEXT NAME OR"
2150 PRINT:PRINT"PRESS ESC TO END INPUT AND TO SORT "
2160 PRINT:PRINT"THE DIRECTORY ."
2170 GETZ$
2180 IFZ$<>CHR$(27)THENNEXTZ
2190 GOTO300
32833 STOP

```

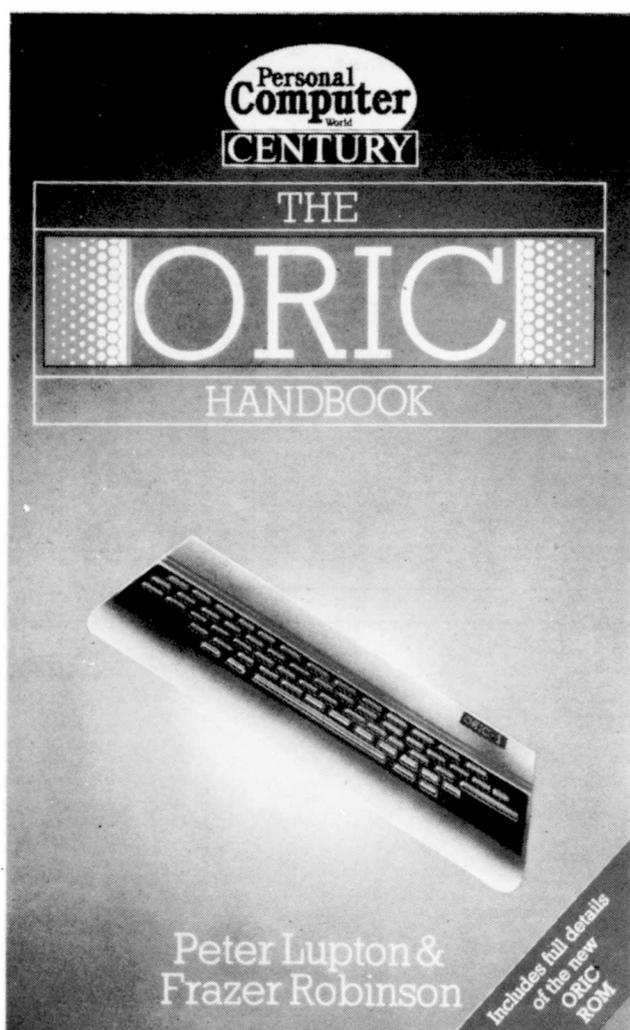
```

1 CLS
2 GRAB
5 GOTO400
100 REM START NEW DIRECTORY
110 DIMA$(300)
120 FORD=1TO300
130 INPUT"NAME";B$
140 INPUT"TELEPHONE NUMBER";C$
150 CLS
160 PRINTB$+" ";C$
170 PRINT:PRINT"IS THIS CORRECT? Y\N"
180 GETZ$
190 IFZ$="N"THENGOTO130
200 A$(D)=B$+" "+C$
210 CLS
220 PRINT"PRESS ANY KEY TO ENTER NEXT NAME OR"
230 PRINT:PRINT"PRESS ESC TO END INPUT AND TO SORT "
240 PRINT:PRINT"THE DIRECTORY ."
260 GETZ$
270 IFZ$<>CHR$(27)THENNEXTD
300 REM DIRECTORY SORTED
305 FORD=1TO300:IFLEFT$(A$(D),1)<CHR$(65)THEN310ELSENEXTD
310 CLS
320 B=0
330 G=D
340 Z=1
345 B=Z+1
350 IFB>GTHEN380
355 IFA$(B)>A$(Z)THEN365
360 Z=Z+1:GOTO345
365 Q$=A$(Z)
370 A$(Z)=A$(B)
375 A$(B)=Q$
376 GOTO360
380 PRINTA$(G)
385 G=G-1
390 IFG>0THENGOTO340
395 PRINT:PRINT"PRESS ANY KEY TO GOTO MENU":GETZ$
400 CLS
405 PRINT" TELEPHONE DIRECTORY"
410 PRINT:PRINT:PRINTSPC(8);CHR$(96)+"
MICHAEL CALDWELL 1983"
420 PRINT:PRINT"PLEASE CHOOSE ONE "
430 PRINT:PRINT"1)START NEW DIRECTORY"
440 PRINT:PRINT"2)TO VIEW DIRECTORY"
450 PRINT:PRINT"3)TO SAVE DIRECTORY ON TAPE"
460 PRINT:PRINT"4)TO SEARCH FOR A NUMBER"

```

TAKE YOUR ORIC TO THE LIMITS -AND BEYOND

The Oric Handbook



PETER LUPTON and FRAZER ROBINSON

The Oric Handbook offers its readers the opportunity of harnessing the power of one of the newest and most exciting microcomputers. A clear step-by-step introduction opens the Oric to the beginner while the wealth of hints and tips, exciting programs and applications makes the book essential reading for even the experienced programmer. There are also lengthy sections on sound and graphic capabilities which are particularly impressive features of the Oric.

The book contains full details of the differences between the old and new ORIC ROM and also documents all the new features and commands.

224pp **£5.95**

*Available through your local bookshop
or if you experience any difficulty
please fill in the form below*

ORDER FORM

To: George Philip Services Ltd,
Arndale Road, Wick, Littlehampton,
West Sussex BN17 7EN

Please send me _____ copy/copies of
THE ORIC HANDBOOK by **PETER LUPTON** and
FRAZER ROBINSON at £6.50 per copy (post paid)

I enclose my cheque/postal order for £6.50 per
copy

Please make payable to George Philip

Name _____

Address _____

Please allow up to 28 days for delivery

Features

INTRODUCTION

This time we feature Chapter 12 from the Oric Handbook – Low Resolution Graphics

LOW RESOLUTION GRAPHICS

In addition to displaying text on the screen, the ORIC can generate and display a variety of graphics shapes and symbols.

There are two distinct types of graphics – Low Resolution and High Resolution. The difference between the two is in the size of the individual blocks, or **pixels**, (picture elements) which make up the display, and in the amount of memory used by the display.

LOW RESOLUTION GRAPHICS

Low resolution graphics use the same screen layout as text mode – 27 rows of 40 columns. In low resolution mode, or Lores, each square on the screen which contains a letter or number in TEXT mode can contain either a letter or number, or a Lores graphics character.

To see the range of Lores characters, type in and **RUN** this program.

```
5 CLS
10 LORES1
20 FOR I=32 TO 95
30 PRINT CHR$(I);SPC(6);
40 NEXT I
```

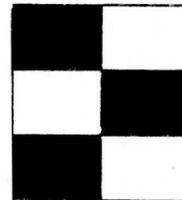
The Lores graphics on the ORIC are of the same type as those used for viewdata services such as PRESTEL, CEEFAX and ORACLE. They are built up from a grid composed of six blocks.

Each block in the character is assigned a number, and these are used to determine the character

1	2
4	8
16	32

The Lores character grid

code. For example, suppose we wanted to display a character which looked like this:



To find the code for this character, first add up the numbers of the shaded squares

$$1+8+16=25$$

Then add 32

$$25+32=57$$

This is the ASCII code for the character '9'.

In order to display our graphics character, we need to put character 9 on the screen. However, we must also specify that we require a graphics character, and not the number '9'. There are two ways to do this.

In **TEXT** mode:

```
Clear the screen (CTRL L)
Move the cursor in from the left with the space bar
Press ESC, then I
Now press 9
```

Our graphics character is displayed.

The ESC I sequence instructed the ORIC that the next character should be the graphics character corresponding to '9', rather than the alphanumeric character.

We can use the same technique within a program:

```
10 CLS
20 PRINT "CHR$(27)" "9"
```

Features

Here we use CHR\$(27) to represent to 'ESC' character because we cannot type the 'ESC' character into a program.

You can use this short program to try out other characters

```
10 CLS
20 INPUT "ASCII CODE";C
30 PRINT C;CHR$(27);"I";CHR$(C)
40 GOTO 20
```

As you can see this method allows both alphanumeric and Lores graphics characters on the same line.

Depending upon when you bought your ORIC, you may find that the LORES characters are a bit lopsided, because the pixels are not all the same size. This is a fault of the ORIC, and to get around it you will either have to design your low resolution displays accordingly, or reprogram the alternate character set. This short program will do the job.

```
10 FOR I=47368 TO 47858
20 IF PEEK(I)=240 THEN POKE I,56
30 IF PEEK(I)=15 THEN POKE I,7
40 NEXT
```

You could include this program as a subroutine in programs that use Lores graphics characters.

There is another way of using Lores graphics, which was used in the first example program – the BASIC command **LORES**.

LORES

LORES is followed by a parameter N, which can be either a 0 or a 1. A 0 means that all characters printed on the screen will be from the standard character set; a 1 selects the alternate set, which contains the 64 Lores characters.

The following program illustrates the difference between the two by first writing a message using the standard set, then switching to the alternate set and rewriting the message.

```
5 CLS:A$="CHARACTER SET DEMO"
10 LORES0
20 PRINT
30 PRINTA$
40 WAIT 500
50 LORES1
60 PRINT
70 PRINT A$
80 WAIT 500
90 CLS
```

Each time a **LORES** command is executed, the screen is cleared to black, and every subsequent **PRINT** or **PLOT** statement is executed using the specified character set, until another **LORES**, **TEXT** or **CLS** statement is executed.

MIXING CHARACTER SETS

It is possible to use characters from both character sets on the screen at the same time, by sandwiching the 'foreign' character or string with CHR\$(8) and CHR\$(9), as in this example:

```
10 LORES0
20 M$="LORES 0 SELECTED"
30 U$=
   CHR$(9)+"#####"
   +CHR$(8)
40 PLOT 12,5,M$
50 PLOT 11,6,U$
60 PLOT 1,9,"UNDERLINING USING
   ALTERNATE CHARACTERS"
```

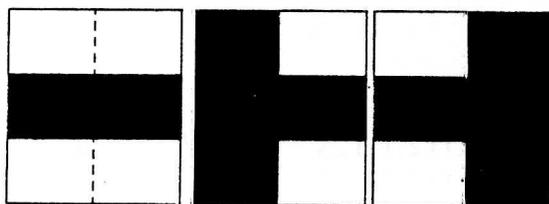
This technique works in reverse: you can display alphanumerics in LORES 1:

```
10 LORES1
20 U$=CHR$(8)+"TEXT in LORES1"
   +CHR$(9)
30 M$="NANANANA"
40 FOR Y=0 TO 24
50 PLOT 16,Y,M$
60 NEXT Y
70 PLOT 12,12,U$
```

As we saw in the previous example, Lores graphics characters can be assembled into strings in the same way as text characters, and we can use this method to draw lines and patterns.

Features

For example, we can draw a grid pattern on the screen using the three characters shown below:



First, find the character codes by adding up the numbers of the shaded squares and adding 32:

$4+8=12$	$12+32=44$	','
$1+4+8+16=29$	$29+32=61$	'='
$2+4+8+32=46$	$46+32=78$	'N'

We build up a string containing the characters ',=N'

A\$=" ,=N"

For use in TEXT mode, the string must begin with a space, followed by the 'ESC' character and an 'ESC' code

Our string becomes

A\$=" "+CHR\$(27)+"I"+",=N"

We can now PRINT the expanded string.

```
10 A$=" "+CHR$(27)+"I"+
    ",=N,=N,=N,=N,=N,=N,=N,=N,=N"
20 PRINT A$
```

ESCAPE CODES

We can obtain double weight and flashing characters with Lores graphics.

If we alter line 10 of our example to read

```
10 A$=" "+CHR$(27)+"M"+
    ",=N,=N,=N,=N,=N,=N,=N,=N,=N"
```

our graphics string can be made to flash.

Similarly, double height graphics can be obtained by replacing the M in line 10 with a K, and adding CHR\$(4) (which is CTRL D) to the string

```
10 A$=" "+CHR$(4)+CHR$(27)+"K"+
    ",=N,=N,=N,=N,=N"+CHR$(4)
```

Remember that double height characters require accurate vertical positioning – to prevent them appearing with the top half of the character below the bottom half, you must position the string on an even numbered line.

To make the double height characters flash, replace the 'K' with an 'O'.

MOVING GRAPHICS

We can use the PLOT command to place graphics characters anywhere on the screen in LORES1 mode. This program creates a string of characters, and moves them slowly across the screen.

```
10 LORES1
20 A$="XU"
30 FORX=1 TO 36
40 PLOTX,10,A$
50 PLOTX-1,10," "
60 WAIT10
70 NEXTX
```

COLOUR WITH LORES GRAPHICS

In previous chapters, we learned how to change foreground and background colours with the INK and PAPER commands.

This works equally well when PRINTing graphics strings in TEXT mode. Since INK and PAPER are global commands, the entire screen will be affected.

However, these commands are not as useful in LORES modes. You may change the colour of the characters from either character set, using INK, in either LORES mode, but PAPER doesn't work: the screen is always set to black.

This short program plots a graphic string in LORES 1 and displays all the INK colours.

```
10 LORES1
20 PLOT15,15,"ABCDEFGG"
30 FOR I=0 TO 7
40 INK I
50 WAIT 100
```

Features

```
60 NEXT I
70 WAIT 100
80 CLS
```

If we change line 40 to read

```
40 PAPER I
```

you will notice that only one of the screen columns adopts the **PAPER** colour – the rest of the screen remains black. This is because of the way in which the ORIC controls its screen display – with the use of attributes. You can make use of attributes in LORES modes in exactly the same way as in TEXT mode.

This program uses most of the ideas in this chapter to plot a histogram (bar chart), showing the spread of numbers generated by the RND command.

```
5 INK7:PAPER0
10 LORES1
20 DIMA%(40)
30 PRINTCHR$(17)
40 PRINTCHR$(20)
50 Y$="5"
60 X$="#"
70 P$="£"
80 T1$=CHR$(8)+"Histogram
  showing Random Numbers" +
  CHR$(9)
90 T2$=CHR$(8)+"between 0 and 35
  being generated"+CHR$(9)
100 N$=CHR$(8)+" 0
  Number
  35"+CHR$(9)
110 GOSUB 500
120 GOSUB 200
130 FOR C=1 TO 300
140 : N=INT(RND(1)*36)
150 : A%(N)=A%(N)+1
160 : IFA%(N)>20 THEN A%(N)=20
170 : PLOT N+3,23-A%(N),P$
180 NEXTC
190 GETZ$: PRINT CHR$(17)
  CHR$(20): CLS
195 END
200 FOR Y=3 TO 23
```

```
210 PLOT 1,Y,Y$
220 NEXTY
230 FORX=1 to 38
240 PLOTX,23,X$
250 NEXTX
260 PLOT2,0,T1$
270 PLOT2,1,T2$
280 PLOT1,24,N$
290 RETURN
500 FORI=48121 TO 48961 STEP 40
510 POKE I,5
520 POKE I+2,3
530 NEXTI
540 RETURN
```

Lines 5 to 40 set the display mode, turn off the cursor and select lower case (simply to tidy up the display).

Lines 50 to 70 set the characters used for the X and Y axes and the bars of the chart.

Subroutine 500 places attributes on the screen to control the colour of the axes.

Subroutine 200 draws the axes and labels the chart.

Lines 130 to 180 plot the results for the function in line 140 for 300 numbers, with line 160 guarding against any part of the histogram overwriting the titles.

SUMMARY

There are two low resolution graphics modes, designated LORES0 and LORES1, which display characters from the standard and alternate character sets respectively on a black screen.

Characters from both character sets can be displayed in either mode by sandwiching the 'foreign' characters with CHR\$(8) and CHR\$(9).

The LORES screen has the same layout as the TEXT screen, and its appearance can be controlled using attributes in the same way as the TEXT screen.

"Chaser"

By R. Titchard



The idea of the game is not unique, you have to move around the screen with the cursor keys, avoiding the trail left by ORIC, and taking care not to hit the sides of the wall, your own Path, or doubling back on yourself. But In my game you have to trick ORIC into a corner – AND do it against the clock!

The skill levels decide how long you have.

The games are in sets of 10. (Best out of 10 wins the set) and the sets can just keep mounting up.

The game can still be improved, by – for instance, arranging a little more SPEED, although the time limits make up for any

luck in this area.

The sound effects are also interesting and may be useful in many other games situations.

The plotting of the screen messages is a very lively routine, which does away with messy PLOT routines.

```

1 REM Roy Titchard:1983: CHASER:
2 PRINTCHR$(20) :PRINTCHR$(6)
3 FORI=0TO7:READ K :FORJ=35TO38
4 POKE46080+I+(J*8),K
5 DATA 1,90,88,15,66,99,0,100
10 GOSUB9000
12 CLS
15 GOSUB4000
20 HI=0
21 ST=0:YO=0
22 POKE618,10
23 CLS:PRINT:PRINT" ";
24 PRINTCHR$(4):CHR$(27);"J";
25 INPUT"WHAT IS YOUR 1ST NAME ";NAME$
26 PRINTCHR$(4):PRINTCHR$(20)
27 TIME=0:WAIT100:GOTO30
30 GOSUB8000
40 AS=KEY$
50 IFA$>"ORAS<CHR$(8)THEN70
60 M=ASC(AS):J=0
70 HI=HI+
(N=11)-(M=10)
75 POKE616,25 :PRINT:POKE617,12 :PRINTCHR$(27);"GTIME ";TIME
76 IF TIME=SK THEN POKE616,25:PRINT:POKE617,12:PRINT"
"
77 TIME=TIME+1
78 IFTIME>SK THEN
TIME=0:GOSUB30000 :
GOSUB
2000
79 GOSUB5000
80 H2=H2+
(M=8)-(M=9)
81 IF TIME=SK THEN PLOT8,23,"
"
82 PLAY0,1,1,240
90 BS=
CHR$(SCRN(H2,HI))
100 PLAY0,1,1,140
110 IFS<<" THENGOSUB2000
120 J=1
130 PLOT8,HI,LEFT$(NAME$,1)
140 U=0
150 IFU=5THENGOSUB1000
160 C3=C1:C4=C2
170 C1=C1+(N=11)-(N=10)
180 C2=C2+(N=8)-(N=9)
190 C5=CHR$(SCRN(C2,C1))
200 IFC5<>" THENN=U+8:C1=C3:C2=C4:U=U+1:GOTO150
220 PLOT2,C1,"#"
240 GOTO40
1000 PING
1010 HS=HS+1
1015 WAIT100
1020 CLS
1025 TIME=0
1030 PRINT"SCORE: COMPUTER ";CS,NAME$;" ";HS
1040 PRINT"-----"
1050 FOR L=1 TO20
|"
1060 PRINT" |"
1070 NEXTL
1075 PRINT"-----"
1080 C1=10:C2=10
1090 HI=10:H2=20
1100 IF HS=10 THEN GOTO 3000
1110 RETURN
2000 PING
2010 CS=CS+1
    
```

```

2015 WAIT100
2020 CLS
2025 TIME=0
2030 PRINT"SCORE: COMPUTER ";CS,NAME$;" ";HS
2040 PRINT"-----"
2050 FORL=1TO20
2060 PRINT"|"
2070 NEXTL
2075 PRINT"-----"
2080 C1=10:C2=10
2090 H1=10:H2=10
2100 IF CS=10 THENCLS:GOTO3000
2110 RETURN
2115 WAIT100
3000 CLS
3001 TIME=0
3002 IFCS=10THENST=ST+1
3003 IFHS=10THENYO=YO+1
3010 IF CS=10 THEN GOSUB10000
3020 IF HS=10 THEN GOSUB20000
3030 PRINT:PRINT:PRINT
3035 PRINT" THE COMPUTER HAS WON ";ST;" SETS":PRINT:PRINT

3036 IFYO=0THENPRINT" YOU HAVE NOT WON A SET YET ";NAME$:GOTO3038
3037 PRINT" YOU HAVE WON ";YO;" SETS ";NAME$
3038 PRINT:PRINT:PRINT:PRINT:PRINT
3039 PRINT"PRESS ANY KEY":GETA$:GOSUB4000:GOTO8000
4000 PRINT:PRINT" ";
4005 PRINTCHR$(4);CHR$(27);"JSKILL LEVEL <1 EASY TO 5 HARD> ";
4010 INPUTSK
4020 IF SK= 1 THEN SK=100
4030 IF SK= 2 THEN SK=90
4040 IF SK= 3 THEN SK=80
4050 IF SK= 4 THEN SK=70
4060 IF SK= 5 THEN SK=50
4070 PRINTCHR$(4)
4075 CLS
4080 RETURN
5000 IFSK=100AND TIME=85THENGOSUB31000
5010 IF SK=90 AND TIME=75 THEN GOSUB31000
5020 IF SK=80 AND TIME=65 THEN GOSUB31000
5030 IF SK=70 AND TIME=55 THEN GOSUB31000
5040 IF SK=50 AND TIME=35 THEN GOSUB31000
5050 RETURN
8000 J=1:C1=10:C2=10
8010 H1=10:H2=20
8020 HS=0:CS=0 :TIME=0
8025 PAPER0
8030 INK INT(RND(1)*6)+1
8040 CLS
8045 TIME=0
8050 PRINT"SCORE: COMPUTER ";CS,NAME$;" ";HS
8060 PRINT"-----"
8070 FORL=1TO20
8080 PRINT"|"
8090 NEXTL
8100 PRINT"-----"
8110 M=9:N=8
8120 RETURN
9000 CLS:PAPER0:INK6:
9002 PING
9005 PRINT:PRINT" ";
9010 PRINTCHR$(4);CHR$(27);"A";CHR$(27);"J          C H A S E R"

```

```

9020 PRINTCHR$(4)
9025 PRINT:PRINT
9027 PRINTCHR$(96);
9028 $$=" Roy Titchard : 1983:"GOSUB9500:PRINT
9030 $$= "IN THIS GAME IT'S YOU AGAINST THE      COMPUTER AND TIME!
":GOSUB9500
9040 PRINT
9045 $$= "THE COMPUTER IS DEFINED AS '#' AND      YOU WILL BE THE"
:GOSUB9500
9050 PRINT
9055 $$= "FIRST LETTER OF YOUR NAME.":GOSUB9500
9060 PRINT
9065 $$= "YOU MOVE AROUND THE SCREEN VIA THE      ARROW KEYS.":GOSUB
9500
9068 PRINT
9070 $$="YOU MUST NOT BUMP INTO YOURSELF OR      THE COMPUTERS PATH":
GOSUB9500
9075 PRINT
9080 $$= "IT IS ALSO FATAL IF YOU DOUBLE BACK      ON YOURSELF.":GOS
UB9500
9081 PING
9082 PRINT:PRINT"  PRESS ANY KEY"
9083 GETA$:CLS:INK2:PAPER0:GOTO9085
9085 PRINT:PRINT:PRINT:PRINT:PRINT
9087 PING
9090 $$= "THERE ARE TEN GAMES TO EACH SET.":PRINT:GOSUB9500
9095 $$= "AND THE SKILL LEVEL CHOSEN DETERMINES THE AMOUNT OF TIME
":GOSUB9500
9105 $$="YOU HAVE TO TRAP ORIC! - SO THINK FAST  OR LOSE FAST!":GOS
UB9500
9110 PING
9115 PRINT:PRINT:PRINT:PRINT:PRINT
9120 PRINT:PRINT
9122 PRINT"          PRESS ANY KEY"
9125 GETA$
9130 PRINTCHR$(20):GOTO12
9500 : FOR I=1 TO LEN($$): PRINT MID$( $$,I,1);:FOR J=1 TO 40 : NEXT

9510 : PLAY1,1,1,20 : SOUND1,3000,0:  PLAY 0,0,0,0
9520 NEXT:PRINT
9530 RETURN
10000 PRINT:PRINT" ";
10005 PRINTCHR$(4);CHR$(27);"N          C O M P U T E R   W I N S"
10010 PRINTCHR$(4)
10020 RETURN
20000 PRINT:PRINT" ";
20005 PRINTCHR$(4);CHR$(27);"N   Y O U   H A V E   W O N   ";NAME$
20010 PRINT CHR$(4)
20040 RETURN
30000 PRINTCHR$(4):POKE616,23:PRINT:POKE617,2:PRINTCHR$(27);"NTIME
PENALTY"
30005 REM
30010 POKE616,23:PRINT:POKE617,20:PRINTCHR$(27);"NTIME PENALTY"
30011 FORJ=1TO300STEP2:PLAY 7,0,0,0:SOUND1,J,10:SOUND2,J*1.2,12:NEX
T
30012 PLAY0,0,0,0:EXPLODE
30015 PRINTCHR$(4)
30020 WAIT300:CLS:RETURN
31000 FORJ=35TO0STEP-2:PLAY7,0,0,0:SOUND1,J,15:SOUND2,J*1.2,15:NEXT

31001 PLOT8,23,CHR$(12)+"TIME PENALTY IMMINENT"
31002 PLAY0,0,0,0
31003 PING
31005 RETURN
31104 SOUND1,550,0:PLAY1,0,1,1600

```

"Chaser" Listing

Software

BATTLESHIPS

By S. W. Lucas

This is a computerised version of the familiar game of BATTLESHIPS. Your opponent is the computer. The computer will hide 5 ships on a 9×9 grid and you must do the same. You will then take turns with the computer to try to bomb the other's ships by firing a missile at a particular co-ordinate.

The player who sinks all of his opponent's ships first is the winner.

The game starts by displaying the instructions, after which you are asked to place your ships on the grid. Co-ordinates are entered as (for example) E5 - note the letter must come first.

You must first give the bow co-ordinate and then the stern coordinate. If the values entered don't make sense, you will be asked to try again: NOTE the LETTER must come first ie. E5 is valid 5E is not!

SHIP SIZES

Aircraft Carrier = 6
Battleship = 5
Cruiser = 4
Submarine = 3
Destroyer = 2

After the board is set up, you and the computer will take turns at trying to destroy each other's ships.

Good Luck.

Variables used

A\$(X,Y) = ships grid
B\$(X,Y) = your grid
other variables are used as counters

Possible changes

1. The ships could be represented as ships rather than letters.
2. The algorithm for destroying the rest of your ship after the first hit could be altered to make the game harder.
3. Removing all REM statements will allow the game to run in 16K machines.

```
2 YC=0:CC=0
10 REM BATTLESHIPS GAME
11 REM THIS GAME CAN BE EASILY MODIFIED TO SUIT THE USER:-SEE REM S
TATEMENTS
12 GRAB:REM RELEASES SPACE OCCUPIED BY HIRES:NOT NEEDED FOR THIS GA
ME WITH
13 REM 48K MACHINE
20 REM BASED ON THE TRADITIONAL GAME
30 REM BY S.W. LUCAS
40 REM AUGUST 1983
45 REM AN ORIGINAL PROGRAM FOR THE ORIC 1 48K
46 REM WILL RUN ON THE ORIC1 16K MACHINE ONLY IF ALL REM STATEMENTS
REMOVED
47 REM DIFFICULTY LEVEL CAN BE ADJUSTED :SEE LINE 25041
50 TEXT
51 DIMA$(9,9),B$(9,9),C$(6)
54 GOSUB10000
55 CLS
60 PAPER7:INK1
70 FOR X=1TO6:READC$(X):NEXTX
80 DATA" " "Destroyer" "Submarine"
85 DATA"Cruiser"
90 DATA"Battleship" "Aircraft Carrier"
100 FORY= 2 TO 20
110 PLOT20,Y,124
120 NEXT
130 FOR X = 1 TO 9
140 PLOTX*2+1,1,CHR$(48+X)
145 PLOTX*2+19,1,CHR$(48+X)
150 NEXT
160 REM PRINT LETTERS DOWN SIDE OF SCREEN
```

BATTLESHIPS

```

170 FORX=1 TO 9
180 PLOT1,X*2+1,CHR$(64+X)
190 NEXT X
200 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
210 PRINT:PRINT:PRINT:PRINT
220 PRINT:PRINT:PRINT
230 PRINT:PRINT:PRINT:PRINT
240 GOSUB11000
250 GOSUB13500
252 REM CC=COMPUTER'S SCORE:YC=YOUR SCORE
260 REPEAT
270 GOSUB16000 :REM YOUR GO
280 GOSUB25000:REM COMPUTER'S TURN
350 UNTIL CC=20 OR YC=20
351 IFCC=20THENAA$="I WIN" ELSEA$="YOU WIN !!!!"
355 CLS:INK4:PLOT1,10,A$
360 PLOT1,15,"DO YOU WANT TO PLAY AGAIN?"
365 GETH$:IFH$(<>"N") THEN RUN
400 END

9900 REM THE END OF THE PROGRAM IS FOLLOWED BY THE SUBROUTINE OF INSTRUCTIONS
9999 END

10000 CLS:PAPER0:INK5
10005 PRINTCHR$(12)
10010 PRINTCHR$(4);CHR$(131);CHR$(27);"          BATTLESHIPS"
10020 PRINTCHR$(4)
10030 PRINT:PRINT:PRINTCHR$(129);"By S. W. Lucas <C> 1983"
10035 PRINT:PRINT:PRINT
10040 PRINT"      This is a computerised version of the traditional game of";
10050 PRINT" BATTLESHIPS."
10060 PRINT"      The game is played on a 9 by 9 grid. Each point on the grid is"
10070 PRINT" identified by a label such as A3, F5 etc."
10080 PRINT"      In this version of the game, your opponent is the computer.";
10090 PRINT" The computer will hide 5 ships on this grid";
10095 PRINT" and you must do the same."
10100 PRINT"      You will take turns with the computer to try and guess the"
10110 PRINT" the co-ordinates of your opponents ships."
10120 PRINT"      The winner is the one who locates the position of a ship";
10130 PRINT" of the opponents ships!";
10135 REM WAIT FOR ANY KEY TO BE PRESSED BEFORE RETURNING TO THE GAME !
10140 GETF$
10145 REM ADD ANY EXTRA INSTRUCTIONS FOR THOSE WHO HAVEN'T PLAYED THE GAME
10146 REM BEFORE HERE
10150 RETURN
11000 LET D= INT(RND(1)*81)
11010 REM THIS SUBROUTINE CHOOSES THE COMPUTERS POSITIONS
11020 REM FOR THE SHIPS ON THE 9 * 9 GRID NOTE 9*9 =81 - LINE 11000
11030 REM THE ARRAY A$(X,Y) HOLDS THE LOCATIONS OF THE SHIPS !
11040 LET A=INT(RND(1)*2)
11045 REM IF A=0 THEN SHIP HORIZONTAL
11046 REM IF A=1 THEN SHIP VERTICAL

```

```

11047 LET X=6 :REM SIZE OF AIRCRAFT CARRIER
11048 LETK=INT(RND(1)*3)+1:REM RANDOM X POSITION OF AIRCRAFT CARRIER
11049 LET L=INT(RND(1)*3)+1:REM RANDOM Y POSITION OF AIRCRAFT CARRIER
11050 GOSUB12000
11055 IF AA=1 THEN 11047 :REM IF LOCATIONS ALREADY OCCUPIED TRY AGAIN
11056 Z$="A" :REM TYPE OF SHIP
11060 GOSUB13000:REM IF LOCATIONS EMPTY PUT SHIP THERE
11070 A=INT(RND(1)*2):REM A=0 :-SHIP HORIZONTAL, A=1:SHIP VERTICAL
11074 LET X=5 :REM SIZE OF BATTLESHIP
11075 LETK=INT(RND(1)*4)+1:REM RANDOM X POSITION OF BATTLESHIP
11077 LET L=INT(RND(1)*4)+1:REM RANDOM Y POSITION OF BATTLESHIP
11080 GOSUB12000
11085 IF AA=1 THEN 11075: REM IF LOCATION ALREADY OCCUPIED, TRY AGAIN
11088 Z$="B":REM TYPE OF SHIP
11090 GOSUB13000:REM IF LOCATION EMPTY, PUT SHIP THERE
11095 A=INT(RND(1)*2):REM SHIP HORIZONTAL OR VERTICAL
11100 LETX=4:REM SIZE OF CRUISER
11105 LET K=INT(RND(1)*5)+1:REM X POSITION OF CRUISER
11107 LETL=INT(RND(1)*5)+1:REM Y POSITION OF CRUISER
11108 GOSUB12000:IFAA=1 THEN 11105:REM TRY AGAIN IF LOCATION OCCUPIED
11110 Z$="C":REM CRUISER
11113 GOSUB13000:REM LOCATE CRUISER
11115 A=INT(RND(1)*2):REM VERT OR HORIZ.
11117 LET X=3:REM SIZE OF SUBMARINE
11120 LET K=INT(RND(1)*6)+1:REM X POS. OF SUBMARINE
11122 LET L=INT(RND(1)*6)+1:REM Y POS. OF SUBMARINE
11125 GOSUB12000:IFAA=1 THEN 11120:REM IF LOCATION ALREADY OCCUPIED, TRY AGAIN
11127 Z$="S":REM SUBMARINE
11130 GOSUB 13000:REM LOCATE SUBMARINE
11133 A=INT(RND(1)*2):REM HORIZ. OR VERT.
11135 LET X=2:REM SIZE OF DESTROYER
11140 LET K=INT(RND(1)*7)+1:REM X POS. OF DESTROYER
11142 LET L=INT(RND(1)*7)+1:REM Y POS. OF DESTROYER
11145 GOSUB12000:IF AA=1 THEN 11140:TRY AGAIN IF LOCATION ALREADY OCCUPIED
11150 Z$="D":REM SHIP TYPE= DESTROYER
11152 GOSUB13000:REM LOCATE SHIP
11160 RETURN
11999 RETURN :REM END OF ROUTINE FOR COMPUTER CHOICE
12000 AA=0
12001 REM SUBROUTINE TO CHECK IF ARRAY ELEMENTS EMPTY
12005 FORP=0 TO X
12010 IF A =0 THEN IF A$(K+P,L)<>" " THENAA=1
12020 IFA=1 THEN IF A$(K,L+P)<>" " THENAA=1
12030 NEXT P
12040 RETURN
13000 FOR P=1 TO X
13010 IF A=0 THEN A$(K+P,L)=Z$
13020 IF A=1 THEN A$(K,L+P)=Z$
13030 NEXT
13040 RETURN
13500 FORP=6 TO 2 STEP-1
13501 K$="Ship size = "+STR$(P)
13502 PLOT1,21,K$

```

Software

BATTLESHIPS

```
13504 AB=0
13505 REM SUBROUTINE FOR PLAYER TO SELECT THE POSITION OF HIS/HER S
HIPS
13510 PLOT1,23,"Enter the bow position of your "
13511 PLOT1,24,C*(P):WAIT40
13512 PLOT1,22," "
13513 PLOT25,24," "
13514 E=25
13515 FORA=1 TO 2:REM GET POINTS OF SHIPS COORDINATES
13517 GETD*(A)
13518 PLOTE,24,D*(A)
13519 E=E+2
13520 NEXT
13522 PLOT25,24," "
13523 GOSUB14000:Y1=Y:X1=X:REM CHECK VALIDITY OF POINT
13524 IFAB<>0 THEN 13501
13525 PLOT1,23,"Enter the stern position of your"
13530 PLOT1,24,C*(P):E=25
13532 FORA=1TO2:REM GET COORDINATES OF STERN POSITION
13535 GETD*(A):PLOTE,24,D*(A):E=E+2
13537 NEXT:WAIT40
13538 PLOT1,22," "
13540 GOSUB14000:Y2=Y:X2=X:REM CHECK VALIDITY OF POINTS
13541 IFY1=Y2THENGOSUB15000:GOTO13545
13542 IFX1=X2THENGOSUB15500:GOTO13545
13543 PLOT1,22,"YOU HAVE MADE A MISTAKE":AB=1:GOTO13510
13545 NEXT
13550 RETURN
14000 LET Y=ASC(D*(1))-64
14005 IFY<10RY>9THENGOSUB14500 NOW:-
14010 LETX=VAL(D*(2))
14015 IFX<1 ORX>9THENGOSUB14500:REM SUBROUTINE TO DISPLAY ERROR
14350 RETURN
14500 PLOT1,22,"INCORRECT COORDINATES :TRY AGAIN":AB=1
14505 RETURN
15000 LETL=X1-X2:IFL<1THENL=X2-X1
15002 LET L=L+1 :REM TO SET CORRECT SHIP SIZE
15005 IFL<>PTHENGOSUB14500:P=P+1:RETURN
15006 IFL=6THENZZ$="A"ELSEIFL=5THENZZ$="B"ELSEIFL=4THENZZ$="C"
15007 IFL=3THENZZ$="S"ELSEIFL=2THENZZ$="D"
15010 REM NEXT CHECK TO SEE IF LOCATION IS ALREADY OCCUPIED
15015 IFX1>X2 THENKK=X1:X1=X2:X2=KK:REM IF COORDINATES WRONG WAY- S
WAP ROUND
15020 FOR S=X1 TO X2:IFB*(S,Y1)<>""THENGOSUB14500:P=P+1:RETURN
15022 NEXTS
15023 REM ABOVE CHECKS IF THE ARRAY B*(X,Y) IS EMPTY
15025 FORS=X1 TO X2:B*(S,Y1)=ZZ$:NEXT
15027 FOR S=X1 TO X2
15028 REM DISPLAY CHOICE
15030 PLOTS*2+1,Y1*2+1,ZZ$
15035 NEXT S
15040 RETURN
15500 LETL=Y1-Y2:IFL<1THENL=Y2-Y1
15502 LETL=L+1:REM TO SET CORRECT SHIP SIZE
15505 IFL<>PTHENGOSUB14500:P=P+1:RETURN
15506 IFL=6THENZZ$="A"ELSEIFL=5THENZZ$="B"ELSEIFL=4THENZZ$="C"
15507 IFL=3THENZZ$="S"ELSEIFL=2THENZZ$="D"
```

```
15510 REM NEXT CHECK TO SEE IF LOCATION IS ALREADY OCCUPIED
15515 IFY1>Y2 THENKK=Y1:Y1=Y2:Y2=KK:REM IF COORDINATES WRONG WAY- S
WAP ROUND
15520 FORS=Y1TOY2:IFB*(X1,S)<>""THENGOSUB14500:P=P+1:RETURN
15522 NEXTS
15525 FORS=Y1 TO Y2:B*(X1,S)=ZZ$:NEXT
15527 FOR S=Y1TOY2
15528 REM DISPLAY CHOICE
15530 PLOTX1*2+1,S*2+1,ZZ$
15535 NEXT S
15540 RETURN
16000 PLOT1,21,"ENTER YOUR MISSILE COORDINATES NOW:-
16010 PLOT1,22," "
16011 E=25
16020 PLOT1,23," "
16030 PLOT1,24," "
16040 PLOT1,25," "
16041 AB=0
16044 FORA=1 TO 2:REM GET COORDINATES OF MISSILE
16045 GETD*(A)
16046 PLOTE,24,D*(A):E=E+2:NEXT
16050 GOSUB14000:REM CHECKS IF POINT VALID
16051 IFAB=1THENWAIT100:GOTO16000
16065 REM CHECK IF ALREADY GUESSED LOCATION
16070 IFA*(X,Y)=""+"THENPLOT1,22,"ALREADY TRIED!":WAIT100:GOTO16000
16075 REM CHECK IF COMPUTER HAS NOTHING THERE
16080 IFA*(X,Y)=""+"THENA*(X,Y)=""+":PLOTX*2+19,Y*2+1,"+":PING:RETURN
16085 REM NOW DISPLAY HIT
16090 PLOTX*2+19,Y*2+1,A*(X,Y):A*(X,Y)=""+":YC=YC+1:EXPLODE
17000 RETURN
25000 IFAZ=0THENX=INT(RND(1)*9)+1
25001 PLOT1,21,"HOLD ON WHILST I THINK OUT MY MOVE "
25002 PLOT1,22," "
25003 PLOT1,23," "
25010 IFAZ=0THENY=INT(RND(1)*9)+1
25015 WAIT100
25020 REM AZ=PREVIOUS GUESS = A HIT
25030 IFAZ=0ANDB*(X,Y)=""+"THEN25000
25032 IFB*(X,Y)<>""ANDB*(X,Y)<>""THENLL$=B*(X,Y)
25035 IFAZ=0ANDB*(X,Y)<>""THENLL$=B*(X,Y):B*(X,Y)=""+":GOTO26000
25040 IFAZ=0ANDB*(X,Y)=""+"THENB*(X,Y)=""+":PLOTX*2+1,Y*2+1,"+":PING:R
ETURN
25041 REM THE FOLLOWING ALGORITHM IS A VERY SIMPLE ONE FOR THE COMP
UTER
25042 REM TO FIND THE REST OF YOUR SHIP IF IT HITS ONCE
25043 REM A MORE SOPHISTICATED ALGORITHM WOULD MAKE THE GAME HARDER
TO WIN!
25044 REM ROUTINE CHOSEN TO MAKE IT EASIER TO PLAY
25050 FORP=1TO9:FORD=1TO9
25060 IFB*(P,D)=LL$THENX=P:Y=D:P=11:D=11:GOTO25080
25070 NEXTD,P
25080 IFF=11THENAZ=0:GOTO25030
25085 AZ=0:GOTO25010
25970 RETURN
26000 PLOTX*2+1,Y*2+1,"+":LETCC=CC+1:AZ=1:EXPLODE:RETURN
```

Doggy Bone

David Reid

Instructions

OBJECT

The object of the game is to get the dog back to his kennel. But first you must manoeuvre him through the woods in search of bones to provide him with energy. Then he must cross the river by jumping onto the moving logs, and then into his kennel on the opposite side.

IN PLAY

You have 3 lives to begin with, and at the start of each frame you have 100 seconds to get four dogs back to their kennels. The dog's energy decreases every 2 seconds, and also when he moves up the screen towards the kennels. To replenish his energy he must eat one of the bones which are scattered through the woods. He must not crash into the trees or fall into the river. If his energy level gets too low, he cannot move towards the kennels, and must go back into the woods to find another bone.

BONUS

When you have filled all the kennels, a bonus is added to your score, depending on how much time is left. Then you move onto the next frame which is more difficult since there are



more trees, the river flows faster, and the logs are shorter.

LIVES

You lose a life if you crash into a tree, fall into the river, or hit the fence beside the kennels. You also lose a life if you run out of energy or time. When there are less than 10 seconds left the clock will start ticking – hurry up!

SCORE

You score 5 points for moving the dog towards the kennels. In the first frame, you score 30

points for every bone eaten, and 50 points for reaching the kennels. This score increases as the difficulty level increases. If you do well enough, you will get to enter your name in the Hi-Score table.

CONTROLS

Z : LEFT
X : RIGHT
' : UP
/ : DOWN

Press RETURN to pause, and then press any key to restart the game.

```

0 RELEASE:HIMEM#94FF:POKE#26A,10
1 REM PROGRAM : Doggy Bone
2 REM AUTHOR : David Reid
3 REM DATE : Sept. 1983
4 CLS:PAPER0:INK7
5 PRINT"Storing data ..."
6 PRINTCHR$(20)
10 GOSUB1000 'Store machine code
20 GOSUB1100 'User graphics
30 GOSUB1300 'Hi-Score table
40 GOSUB1400 'Initialize values
50 GOSUB1500 'Print title screen
60 GOSUB1700 'Print frame number
70 GOSUB1800 'Play tune
80 GOSUB2000 'Set up screen
85 GARBAGE=FRE("")
90 DOKE#276,10000:C=0 'Start clock
100 REPEAT
110 DN&(0)GOTO130,140,150,160,170
120 GOTO210
130 PLOT0X,OY,"A":NX=OX-1:GOTO180
140 PLOT0X,OY,"B":NX=OX+1:GOTO180
150 PLOT0X,OY,"C":NY=OY-1:IF E<6 THENNY=OY:GOTO210
155 GOTO180
160 PLOT0X,OY,"D":NY=OY+1:GOTO180
170 PT=DEEK(#276):GETA$:DOKE#276,PT
175 GOTO210
180 GOSUB2300 'Check new position
190 GOSUB2800 'Move dog
195 C=C+2
200 IFNOTOK%THENGOSUB2700 'Dead
210 C=C+1:IFC<10-LE THEN 270
220 C=0:CALLA
230 IFOY<>6THEN250
240 OX=OX+1:IFOX=35THENGOSUB2700
245 NX=NX+1
250 IFOY<>7THEN270
260 OX=OX-1:IFOX=2THENGOSUB2700
265 NX=NX-1
270 GOSUB3500 'Update clock & energy
280 UNTILFALSE
1000 READA :REM 48K Dric version
1001 REPEAT
1002 READD$
1003 REPEAT
1004 V=VAL("#"+D$)
1005 POKEA,V:A=A+1
1006 READD$
1007 UNTILD$="FF"
1008 READA
1009 UNTILA=#FFFF
1010 DOKE#2F5,#400
1011 DOKE#2FC,#9700
1012 A=#9500:RETURN
1020 DATA#400
1021 DATA 20,96,D9,AC,F8,02,C8,8C
1022 DATA 69,02,A5,1F,A4,20,85,12
1023 DATA 84,13,A9,3B,20,DB,CF,4C
1024 DATA 61,CB,FF
1025 DATA#9500
1026 DATA A9,C3,85,66,A9,BC,85,67
1027 DATA A0,00,B1,66,85,68,C8,B1
1028 DATA 66,88,91,66,C8,C0,21,D0
1029 DATA F5,A5,68,91,66,A9,9B,85
1030 DATA 66,A0,21,B1,66,85,68,88
1031 DATA B1,66,C8,91,66,88,C0,00
1032 DATA D0,F5,A5,68,91,66,60,FF
1033 DATA#9700
1034 DATA A0,00,AD,08,02,10,0C,C8
1035 DATA D9,15,97,F0,06,C0,05,D0
1036 DATA F6,A0,00,4C,FD,D3,AA,B0
1037 DATA BB,9F,AF,FF,#FFFF

```

```

1100 FORI=65T082
1101 READD$
1102 FORJ=1T015STEP2
1103 V=VAL("#"+MID$(D$,J,2))
1104 F=#B000+I*8+(J-1)/2
1105 POKEF,V
1106 NEXTJ,I
1107 RETURN
1200 DATA 000008310E0A0A00
1201 DATA 000004231C141400
1202 DATA 0404110E040E1100
1203 DATA 00110E040E110404
1204 DATA 003F0014080A0317
1205 DATA 00040A1D170E0403
1206 DATA 00000203041B0800
1207 DATA 003F100908103F00
1208 DATA 003F003006003F00
1209 DATA 003C023101023C00
1210 DATA 000A04150A040A11
1211 DATA 0000000000000102
1212 DATA 00000000C12210000
1213 DATA 00000000000002010
1214 DATA 040B120A3A0A0A3E
1215 DATA 003F000000000000
1216 DATA 083412141714141F
1217 DATA 000000123F12123F
1300 DIMHI(8),HI$(8)
1301 FORI=1TO8
1302 HI(I)=(11-I)*100:HI$(I)="Dric"
1303 NEXTI:RETURN
1400 S=0:F=0:L=3
1410 T=100
1411 E=30:OX=19:OY=24
1412 NX=OX:NY=OY:NS=32
1413 RETURN
1500 CLS:INK6:PAPER0
1501 A$=CHR$(10)+"Doggy Bone":PLOT9,3,A$:PLOT9,4,A$
1502 PLOT13,7,CHR$(3)+"By D. Reid"
1503 PLOT11,10,CHR$(4)+"For 'Dric Owner'"
1504 PLOT14,17,CHR$(1)+"CONTROLS"
1505 PLOT13,19,CHR$(2)+"LEFT <Z>"
1506 PLOT13,20,CHR$(3)+"RIGHT <X>"
1507 PLOT13,21,CHR$(4)+"UP <^>"
1508 PLOT13,22,CHR$(5)+"DOWN <v>"
1509 PLOT13,23,CHR$(6)+"PAUSE <ret>"
1510 PLOT04,25,CHR$(7)+"Press <return> to start game .."
1511 K$=KEY$:I=0
1512 REPEAT:R=RND(1):I=I+1
1513 UNTIL(KEY$=CHR$(13))OR(I>1000)
1514 IFI>1000 THEN PING:GOTO 1511
1515 RETURN
1700 CLS:PAPER0:INK6:F=F+1:H=0
1701 LE=F+1:IFLE>8 THEN LE=8
1702 A$=CHR$(14)+"Doggy bone"
1703 PLOT13,7,A$:PLOT13,8,A$
1704 A$=CHR$(LEOR1)AND7)+CHR$(10)+"Frame"
1705 PLOT12,13,A$:PLOT12,14,A$
1706 A$=CHR$(4)+CHR$(10)+"Score"
1707 PLOT12,17,A$:PLOT12,18,A$
1708 '23,13;F:'23,14;F
1709 '23,17;S:'23,18;S
1710 RETURN
1800 T$=""
1801 IFF=1THENT$=T$+"5885851&6AA6A1&8CC8C8ACD&8&5&3&"
1802 T$=T$+"5885851&6AA6A1&8CC8C8ACD8531&"
1803 TE=S:OC=3:LT=LEN(T$)
1815 FORI=1TOLT:NT=VAL("#"+MID$(T$,I,1)):O=0C
1816 IFNT>#CTHENO=OC+1:NT=NT-#C
1817 IFNT=0THEN1825
1818 MUSIC1,0,NT,11:MUSIC2,0-1,NT,7
1819 PLAY0,0,0,0
1820 PLAY3,0,1,3000

```

Doggy Bone Listings

Software

```

1825 WAITTE
1826 NEXTI
1827 PLAY0,0,0,0
1828 GARBAGE=FRE("")
1829 RETURN
2000 TEXT:CLS:INK0:PAPER0:CALL#E6CA
2005 PLOT1,0,CHR$(7)+"Setting up ... "
2010 FORI=0T03:POKE48040+40#I,8:NEXT
2015 FORI=4T024:POKE48040+40#I,9:NEXT
2020 FORI=25T026:POKE48040+40#I,8:NEXT
2025 FORI=2T024:PLOT36,1,16:NEXT
2030 PLOT4,4,"LMN LMN LMN LMN"
2035 PLOT2,5,"RROPORRRRRROPORRRRRROPORRRRRROPORRR"
2040 PLOT2,6,"H":PLOT2,7,"H":PLOT20,6,"H":PLOT20,7,"H"
2045 FUK1=11011 LE:PLU12+1,6,"I":PLU12+1,7,"I":PLU120+1,6,"I"
2046 PLOT20+1,7,"I":NEXT
2050 PLOT2+I,6,"J":PLOT2+I,7,"J":PLOT20+I,6,"J":PLOT20+I,7,"J"
2051 R=RND(1)*16+1
2055 FORI=1TOR:CALLA:NEXT
2060 FORI=8T023:FORJ=1TOLE#2+1
2061 R=RND(1)*29+4:PLOTINT(R),I,"F"
2062 NEXTJ,I
2063 R=INT(RND(1)*5)
2065 FORI=1T010+R:R1=RND(1)*15+8
2066 R2=RND(1)*31+3:PLOTINT(R2),INT(R1),"G"
2070 NEXTI
2075 PLOT0,1,CHR$(3)+CHR$(16)
2080 FORI=2T05:PLOT1,I,CHR$(22):NEXT
2085 FORI=6T07:PLOT0,I,CHR$(7)+CHR$(20):NEXT
2090 FORI=8T023:PLOT1,I,CHR$(18):NEXT
2095 PLOT0,24,CHR$(4)+CHR$(23)
2100 PLOT0,25,CHR$(5)+CHR$(16)
2105 PLOT0,26,CHR$(6)+CHR$(16)
2110 PLOT2,25,"Frame":PLOT9,25,CHR$(6)+"Score":
      PLOT20,25,CHR$(2)+"Hi-Score"
2115 PLOT30,25,CHR$(4)+"Lives"
2120 PLOT10,26,CHR$(7):PLOT22,26,CHR$(3):
      PLOT32,26,CHR$(5):PLOT32,1,CHR$(16)
2130 E$=""
2135 FORI=1T05:E$=E$+CHR$(17):NEXT
2140 FORI=1T010:E$=E$+CHR$(19):NEXT
2145 FORI=1T015:E$=E$+CHR$(18):NEXT
2150 PLOT2,0,"":PLOT0,0,CHR$(6)
2155 PLOT2,0,"Energy":PLOT32,0,"Time"
2160 PLOT2,1,E$
2165 !32,1;T:!3,26;F:!10,26;S:!22,26;HI(1):!32,26;L
2170 PLOT0X,OY,"A"
2175 CALL#EB04
2180 RETURN
2300 OS=NS:NS=SCRN(NX,NY)
2305 OK%=(NS=71)OR(NS=72)OR(NS=73)OR(NS=74)OR(NS=80)
2310 IF OK%THENRETURN
2315 IFNS=32THENOK%=(NY<>6)AND(NY<>7)
2320 IFNX<3THENNX=3:OK%=TRUE:NS=32
2325 IFNX>34THENNX=34:OK%=TRUE:NS=32
2330 IFNY>24THENNY=24:OK%=TRUE:NS=32
2335 RETURN
2700 PLOT0X,OY,"K"
2701 GARBAGE=FRE("")
2705 FORI=1T0500STEP8:SOUND1,I,12:PLAY1,0,1,1000:NEXTI:PLAY0,0,0,0
2710 PLOT0X,OY,NS
2715 L=L-1:!32,26;L:PRINTCHR$(30)
2720 IFL=0THENGOSUB3600 "Game over
2730 IF(T<=1)OR(T>100)THENGOSUB1410 ELSE GOSUB1411
2735 !32,1;T:PLOT2,1,E$:PRINTCHR$(30)
2740 PLOT0X,OY,"A"
2745 POP:PULL
2750 IF DEEK(#276)>10000THENGOTO90ELSEGOTO100
2800 DS=SCRN(OX,OY)
2802 PLOT0X,OY,OS:OS=SCRN(OX,OY)
2805 PLOTNX,NY,DS
2810 IFNY>OYTHEN2830

```

```

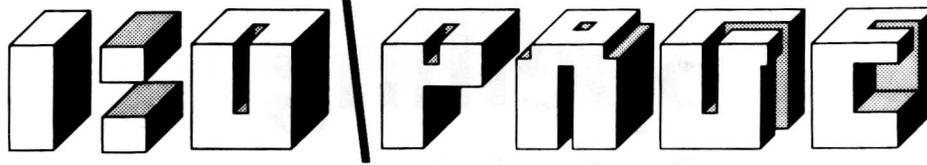
2815 S=S+5:E=E-1:!10,26;S:PRINTCHR$(30):PLOT2+E,1,16
2820 IFE<=0THEN2700
2830 OX=NX:OY=NY
2840 IFNS<>71THEN2860
2845 S=S+20+10#F:E=30
2850 FORI=50T010STEP-5:SOUND1,I,7:PLAY1,0,1,1000:NEXTI:PLAY0,0,0,0
2855 !10,26;S:PRINTCHR$(30):PLOT2,1,E$:NS=32
2860 IFNS<>80THENRETURN
2865 PLOTNX,NY,"E"
2870 FORI=25T050STEP-10:SOUND1,I,10:PLAY1,0,1,1000:NEXTI:PLAY0,0,0,0
2880 S=S+30+20#F:!10,26;S:PRINTCHR$(30):H=H+1
2890 IFH<4THEN2940
2900 T$="EFDECDBCAB9A8978675645342312C1&"
2910 TE=3:OC=3:LT=LEN(T$):GOSUB1815
2920 S=S+T*10:!10,26;S:!32,1;"00":PRINTCHR$(30)
2925 WAIT90:GOSUB1410
2930 POP:PULL:GOTO60
2940 GOTO2730
3000 CLS:PAPER0:INK7
3001 GARBAGE=FRE("")
3005 A$=CHR$(6)+CHR$(10)+"Doggy Bone "+CHR$(5)+"Hi-Scores"
3010 PLOT5,1,A$:PLOT5,2,A$
3015 FORI=1T08:J=I:IFJ>6THENJ=J-5
3020 Z=I*2+3:PLOT6,Z,CHR$(J)+">>"
3025 !7,Z;I:!12,Z;HI(1):PRINTCHR$(30)
3030 PLOT21,Z,HI$(I):NEXT
3035 RETURN
3200 PLOT6,22,CHR$(6)+"Please enter your name"
3205 PRINTCHR$(30)
3206 FORI=1T022:PRINT:NEXT
3207 K$=KEY$
3210 POKE#26A,3
3211 PRINTCHR$(20)
3215 INPUTN$
3216 PRINTCHR$(20)
3220 POKE#26A,10:PRINTCHR$(30)
3225 IFLEN(N$)>16THENN$=LEFT$(N$,16)
3230 HI(8)=S:HI$(8)=N$
3235 FORI=7T01STEP-1
3240 IFHI(1)<HI(I+1)THENGOSUB3300
3245 NEXTI
3250 GOSUB3000
3255 RETURN
3300 Z=HI(I):HI(I)=HI(I+1):HI(I+1)=Z
3305 A$=HI$(I):HI$(I)=HI$(I+1):HI$(I+1)=A$
3310 RETURN
3500 T%=INT(DEEK(#276)/100)
3505 IFT%>=INT(T)THENRETURN
3510 T=T%:!32,1;T:PRINTCHR$(30)
3520 IFT/2=INT(T/2)THENE=E-1:PLOT2+E,1,16
3540 IFT<=0ORE<=0THEN2700
3550 IFT<11THENCALL#FAFA
3555 C=C+1
3560 RETURN
3600 POP:PDP:PULL:K$=KEY$
3601 GARBAGE=FRE("")
3605 PLOT13,3,CHR$(12)+"GAME OVER"
3610 WAIT200
3615 GOSUB3000
3620 IFS>HI(8)THENGOSUB3200
3625 PLOT6,22,CHR$(3)+"Press <1> for a new game"
3630 PLOT12,23,CHR$(5)+"<2> to finish"
3635 REPEAT:GETK$:UNTIL(K$="1")OR(K$="2")
3640 IFK$="1"THEN40
3650 POKE#26A,3:CLS
3660 PAPER7:INK0:PRINTCHR$(20)
3670 END
3680 REM Program by David Reid

```

Doggy Bone Listing

Software

Regulars



Thank you for a nice magazine. However I would like to make some comments on different items in issue 2.

Regulars (Disaster Area)

The TAB function does only work correctly to the first TAB position (the reduction of 13 taken into consideration). It is impossible to use the TAB function to get correctly lined columns because the following TAB positions are dependent of how many characters a field or number consists of. Only cumbersome calculations to calculate number of spaces after having gone over strings can overcome this. Or am I wrong?

Cassette handling

Very interesting items. Would like to see solutions for MERGING and VERIFYING tapes. By the way, Saving the screen works OK but it is impossible to continue working after a reloading of the screen content. I always get the fault report: OUT OF MEMORY, and I can't get around it.

From Bits to Screen Attributes

Has the author really tried his examples on a machine, or has he only read the manual?

(a) As address 48560 refers to column one, you can't POKE

characters into that column as described on page 27 columns 2 and 3.

(b) I can PLOT and POKE characters with numbers greater than 128 and get reversed video, but I can't PRINT. I have tried on several ORICs with the same result, so it must be a bug.

(c) You can't use the example `PRINT CHR$(4); CHR$(27); "O"; "HELLO"; CHR$(4)` because the attribute goes into column 1 (I think). You have to put in one space before the escape character. And the attribute should be N, not O as shown in the example. The following statement works better: `PRINT CHR$(4); " "; CHR$(27); "N"; "HELLO"; CHR$(4)`

Lennart Arndtsson: Uppsala Sweden.

Editor: You have pointed out an important shortcoming in the Oric Owner which we are urgently trying to rectify. This is the problem of accurately proof-reading and testing program listings. With this in mind, we are trying to recruit an additional member of staff to help with the editorial side of the magazine. If you are interested, drop me a line.

Spelling

By S. W. Lucas

An educational program for pupils from 7 upwards.

Aims

This program was written as a game to help pupils to recognise the correct spelling of words.

Description

The program is meant for use within a classroom situation, although changing the messages will make it suitable for use at home. It was developed for use with 8–10 year olds, who have a good vocabulary. It is easy to modify to suit children from 5–9 by changing only a few lines (see below.).

The program will firstly ask the teacher how many pupils are to 'have a go' at this program. Each pupil will be presented with ten questions chosen at random from 60 DATA lines. It is possible to increase the possibilities by adding extra data items and changing line 90 to `AD=INT(RND(J)* number of lines of data-9)`. You will also need to change LINE 1300 to `FOR Z=1 to number of lines of data` and in addition you need to dimension the arrays (line 12) accordingly.

The program will then give each pupil 10 questions chosen from the data items in order. The data pointer is set by line 90 to a



random question. After 10 questions, the pupil will be given their score and the next pupil will be asked to 'have a go'.

When all the pupils have taken their turn to try the program, the results will be printed out either on a printer, if one is available, or on the screen. It is better to use a printer because if more than 23 pupils are listed, the screen will scroll!

Modifications

In order to make this program suitable for pupils of different ages, it needs to be adapted. I have made this a very easy process (as can be seen from the listing).

Lines 10000–10620 contain data.

Because the ORIC 1 will not accept lines of greater than 80 characters in length, it has been necessary to split some data statements into two lines.

The data consists of 4 alternative spellings of the word followed by the number of the correct answer and finally a clue (often a dictionary definition of the word.)

Thus changing the program involves only modifying the DATA lines and (perhaps) changing line 220. Line 220 is a wait statement which should be increased for younger pupils in order to give them more time to read the four alternative answers.

Notes

REM statements are included throughout the program to help the user to adapt the program for their own use. As it stands, the program is ideally suited for pupils from 9 upwards, although the wait statement in line 220 will need to be increased for children who can't read quickly!

```

5 REM SPELLING QUIZ
6 REM AN EDUCATIONAL PROGRAM BY
7 REM S.W. LUCAS
8 J=1:REM SEED FOR RANDOM NUMBER
10 TEXT:GRAB:PAPERO:INK1
11 REM DIMENSION THE ARRAYS IN THE NEXT LINE TO SUIT YOUR USE !
12 DIMA$(61,4),A(61),B$(61)
13 REM A$ HOLDS THE FOUR ALTERNATIVE ANSWERS
14 REM A HOLDS THE NUMBER OF THE CORRECT ANSWER
15 REM B$ HOLDS THE MEANING/CLUE OF THE WORD
16 PRINTCHR$(17):REM TURNS CURSOR OFF
20 GOSUB1000:REM TITLES
25 RESTORE
30 REM NOW READ THE DATA FOR THE QUESTIONS INTO THE ARRAY
31 REM IT IS STRAIGHTFORWARD TO CHANGE THE DATA TO SUIT THE AGE OF
  THE CHILD!
32 REM USE SIMPLER WORDS FOR YOUNG CHILDREN AND EXTEND THE VOCAB FOR
  OLDER
33 REM PUPILS: REMEMBER FOUR ALTERNATIVES,FOLLOWED BY THE NUMBER OF
  THE
34 REM CORRECT ANSWER : FOLLOWED BY THECLUE/WORD DESCRIPTION
35 REM FOR OLDER PUPILS ONLY GIV! THE CLUE, FOR YOUNGER PUPILS GIVE
36 REM A MEANING OF THE WORD : THE WORDS GIVEN ARE ONLY AN EXAMPLE
37 REM CHANGE THE WORDS TO SUIT THE AGE GROUP BEING CONSIDERED
40 GOSUB1300
50 CLS
60 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT"How many children are there"
  :
70 INPUTAA
71 DIMSC(AA):REM ARRAY SC(AA) HOLDS EACH PUPILS SCORE
80 CLS:FORX=1TOAA
82 SC(X)=0:REM SET SCORE TO ZERO
90 AD=INT(RND(J)*51):IFAD<1THENAD=1
91 IFAD>50THENAD=50
95 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
96 PRINT
100 FORXP=1 TO 10:REM SET TEN QUESTIONS
105 REM NOW CHANGE THE ATTRIBUTES SO THAT WORDS PRINTED IN BACKGROU
  ND COLOUR
106 PLOT1,4,0:PLOT1,8,0:PLOT1,12,0:PLOT1,16,0
110 PLOT2,4,A$(AD,1)
120 PLOT2,8,A$(AD,2)
130 PLOT2,12,A$(AD,3)
140 PLOT2,16,A$(AD,4)
144 PLOT1,1,3:PLOT1,2,3:PLOT1,24,6
150 PLOT2,1,"press <space bar> when the correct"
160 PLOT2,2,"spelling of the word is shown"
170 PLOT2,24,B$(AD)
185 PP=0
190 REPEAT
200 PLOT1,PP,5:REM SET ATTRIBUTE TO MAGENTA --- CHOOSE YOUR OWN COL
  OUR
220 WAIT50:REM CHOOSE WAIT TIME TO SUIT PUPILS AGES
221 PLOT1,PP,0:REM SET TO BACKGROUND
222 PP=PP+4:REM TO SET NEXT TO ALTERNATIVEANICOL
225 IFPP>16 THENPP=4
300 UNTIL KEY$=""
310 LETDX=PP/4-1:IFDX=0THENDX=4:REMDX=NUMBER OF ANSWER SELECTED
320 IFA(AD)=DXTHEN GOSUB11000 ELSE GOSUB 12000:REM ROUTINES FOR COR
  RECT OR
321 REM INCORRECT ANSWERS
325 AD=AD+1
326 REM INCREMENT'S THE QUESTION SET
330 CLS:NEXTXP:REM NEXT QUESTION
340 CLS:PLOT1,1,1:PLOT2,1,"What is your name"
345 PRINT:PRINT:PRINT:PRINT:PRINT
350 INPUTN$(X)
360 CLS:PING:PLOT1,10,3
370 PLOT2,10,"Thank you for playing "
380 PLOT1,14,3:PLOT10,14,N$(X)
390 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
392 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
395 PRINTCHR$(131)"You scored :- ";SC(X);" out of ten !"
396 PRINT:PRINT:PRINT
397 IFX<>AATHENPRINT"NOW LET THE NEXT PUPIL HAVE A GO":GOTO399
398 PRINTCHR$(133);"PLEASE CALL THE TEACHER"

```

```

399 PRINT:PRINT:PRINTCHR$(129)" Press <SPACE BAR> when ready"
400 GETS$
490 CLS
500 NEXTX:REM NEXT CHILD'S GO
501 REM NOW THE ROUTINE TO PRINT THE RESULTS
510 CLS:PRINT:PRINT:PRINTCHR$(131)"Do you have a printer attached
  ?"
520 REPEAT: AA$=KEY$:UNTILAA$="Y" ORAA$="N"
530 IFAA$="Y"THEN GOSUB20000 ELSE GOSUB 30000
540 PRINT:PRINT"PRESS <SPACE BAR>"
550 GETD$
560 CLS:PLOT1,10,"Do you want to play again?"
570 REPEAT:GET AA$:UNTILAA$="Y"ORAA$="N"
580 IFAA$="Y"THENRUN
999 END
999 REM TITLES SUBROUTINE
1000 CLS:FORX=1TO2
1010 PLOT1,X,14:PLOT2,X,X+2:PLOT10,X,"SPELLING QUIZ"
1020 NEXT
1030 FORX=10TO15:PLOT1,X,X-9:NEXT
1040 PLOT2,10,"a game for the ORIC 1"
1050 PLOT2,12,"by S. W. Lucas"
1060 PLOT2,14,"The game keeps the score of up to 50"
1070 PLOT2,15,"players (such as a class group)"
1080 PLOT2,16,"and you can get the results printed"
1090 PLOT1,17,5
1100 PLOT2,17,"out later. You will first be asked "
1110 PLOT1,18,2
1120 PLOT2,18,"how many pupils there are."
1130 FORX=20TO25:PLOT1,X,X-19
1140 NEXT
1150 PLOT2,20,"Each pupil will then presented with"
1160 PLOT2,21,"10 questions chosen at random."
1170 PLOT2,22,"Choose your answer by"
1180 PLOT2,23,"pressing the <SPACE BAR>"
1185 PLOT5,25,"PRESS <SPACE BAR> TO CONTINUE"
1190 REPEAT:J=J+1:A$=KEY$: UNTIL A$=""
1191 CLS:PLOT1,10,"please wait whilst I set the questions"
1220 RETURN
1300 FORZ=1TO60
1310 FORY=1TO4
1320 READA$(Z,Y)
1330 NEXTY:READA(Z),B$(Z)
1340 NEXTZ
1350 RETURN
2299 REM ROUTINE TO READ DATA INTO THE ARRAYS
10000 REM DATA FOR THE QUESTIONS
10010 DATA capital,kapital,capitol,capertal,1,of chief importance
10020 DATAdisappointing,disappointing,disapointing,dissappointing,2
10021 DATAnot up to expectation
10030 DATAwhipet,whippet,whippet,wippet,3, a type of dog
10040 DATAttransparent,transparent,transparant,transperant,2
10041 DATA can be seen through
10050 DATAnececity,necesity,necessety,necessity,4,something which i
  s needed
10060 DATAasasination,assassination,assasination,asassination,2
10061 DATAmurder
10070 DATABronkitis,bronchitis,bronchitus,broncitis,2,an illness
10080 DATAcentenry,centenary,centanery,sentenary,1,hundredth anniv
  ersary
10090 DATAcentrifugal,centrafugal,centrifugul,centrifugal,4
10091 DATAforce of a spinning object
10100 DATAconventional,convensional,conventionul,konvensional,1
10101 DATAtthe normal response
10110 DATAallocation,aloccaton,allocation,allocatian,3,to assign
10120 DATAcompulsory,compulsary,compulsery,compulsory,1
10121 DATAsomething you have to do
10130 DATAcontroler,controllor,controlor,controller,4,person in cha
  rge
10140 DATAcalendar,calender,callendar,callender,1,table of the year
  s dates
10150 DATAAppreciate,apreciate,apreciete,appresiate,1,set a high va
  lue on
10160 DATAstyle,styal,stial,stile,1,design
10170 DATAsterilise,steralise,steralize,sterilize,4,to get rid of m
  icrobes

```

Spelling Listing

Software

```

10180 DATAindicate,syndicate,syndecate,sindecate,2,group of people
10190 DATAtemperature,temperature,tempereture,temprature,2,degree of
      heat
10200 DATAidenticle,identacle,identical,identecal,3,alike
10210 DATAhidrogen,hydrngen,hydrogen,hydrogan,3,a chemical element
10220 DATAbugerigar,bugarigar,budgerigar,budgeregar,3,a type of bir
      d
10230 DATAadminister,adminster,adminester,adminestar,1,look after a
      ffairs
10240 DATApharmacy,pharmacry,pharmercy,pharmacy,4,chemists shop
10250 DATApossession,posesion,possession,possession,1,own
10260 DATAretaleate,retaliatate,retalayate,reataliatate,2,get your own
      back
10270 DATAstomache,stumach,stomacke,stomach,4,part of the body
10280 DATArhythm,rithm,rhythem,rhythm,4,part of music
10290 DATAsuccessful,succesful,successful,sucesful,3,do it correctly
10300 DATAsubstansial,substanshal,sustantial,substantiel,3,not inco
      nsiderable
10310 DATAspatious,spachious,spasious,spacious,4,plenty of room
10320 DATAoxigen,oxegen,oxejun,oxygen,4,a gas
10330 DATAorchid,orkid,orchyd,orcid,1,a flowering plant
10340 DATAnavigation,navigashion,navigatian,navigaton,1,sail a ship
      on course
10350 DATAnecessary,neccessary,necessary,neccesary,3,indispensable
10360 DATAmechanical,mechanical,mechanicle,mechanicol,2,works by mac
      hines
10370 DATAimpossibal,impossible,imposible,impossable,2,not allowed
10380 DATAidolise,idolize,idlise,idlize,2,to love
10390 DATAhalucination,hallucinatian,hallucination,halucinatian,3,i
      llusion
10400 DATAforcable,forcible,forsable,forsible,2,done using force
10410 DATAextraction,extracshion,extraktion,extracsion,1,take out
10420 DATAenvelope,henvelope,envylope,envalope,1,you put letters in
      it
10430 DATAendles,hendles,endless,endless,4,without an end
10440 DATAcomputer,computar,computer,computur,1,a machine for game
      s and maths
10459 DATAsituation,situation,situashion,situachion,1,position you
      are in
10460 DATAassemble,assemble,asemble,asembel,2,bring together
10470 DATAaccumalate,accumalate,accumerlate,accumulate,4,heap up
10480 DATAcontent,contant,contente,kontent,1,satisfied
10490 DATAexpreshun,expresion,expression,hexpression,3,wording or p
      hrase
10500 DATAencountar,encoutur,encountter,encounter,4,close contact
10510 DATAmanual,manuel,manuarl,manur1,1,done by hand
10520 DATAflasching,flashzing,flachsing,flashing,4,a lamp turning o
      ff & on
10530 DATAamplifier,amplifier,ammplyfier,amplyfier,2,makes louder
10540 DATAwhasteful,wastfull,wastefull,wasteful,4,not economical
10550 DATAfigure,fighure,phigure,phighre,1,shape
10560 DATAconclider,consider,considure,concidur,2,contemplate
10570 DATAvibrasion,vibration,vibrasian,vabratian,2,move continuous
      ly
10580 DATAaltering,altaring,alturing,haltering,1,changing
10590 DATAprobablie,probably,probably,proably,3,most likely
10600 DATAcomplicated,complhicated,complecated,complacated,1,involv
      ed
10610 DATAcasette,cassette,casete,cassete,2,type of tape
10620 DATAcontoled,kontrolled,controllad,controlled,4,run by
11000 CLS:REM ROUTINE FOR CORRECT RESPONSE
11010 PLOT10,10,"C O R R E C T !"
11020 FORPK=1TO 7
11030 ZAP:PLOT1,10,PK
11040 WAIT20:NEXT:REM CREATES SOUND AND CHANGES THE COLOUR
11049 REM INCREMENT SCORE SC(X) OF XthPUPIL
11050 SC(X)=SC(X)+1
11060 RETURN
12000 CLS:REM ROUTINE FOR INCORRECT ANSWER
12010 PLOT10,1,6:PLOT11,1,"W R O N G"
12020 PLOT1,10,3:PLOT2,10,"It was :-"
12030 PLOT20,10,A*(AD,A(AD)):REM CORRECT SPELLING
12039 REM SOUND EFFECT FOR LOSING
12040 FORXX=1TO6
12045 FORYY=1 TO 150 STEP 3
12050 SOUND1,YY,15

```

```

12060 NEXTYY,XX
12065 SOUND1,0,0
12070 RETURN
19999 REM ROUTINE TO SEND RESULTS TO PRINTER
20000 LPRINT"NAME ", "SCORE OUT OF TEN"
20010 FORX=1TOAA
20020 LPRINTN$(X),SC(X)
20030 NEXT
20040 RETURN
30000 PRINT"NAME", "SCORE OUT OF TEN"
30010 FORX=1TOAA
30020 PRINTN$(X),SC(X)
30030 NEXT
30040 RETURN

```



Spelling Listings

Software

Oric Quickies

Prime Numbers

By E. Hollister

Programs to calculate and print Prime Numbers are varied in their methods of approaching the problem and the time taken to complete the task. The BASIC program listed below will calculate and print (to VDU) Prime Numbers up to the value allocated to 'B' in line 10. Taking 12.5 seconds to run with 'B' set to 1000. Lines 5 and 115 provide a useful on screen check of the program running time and can be applied to other programs.

```
4 CLS
5 DOKE630,65535
10 B=1000
20 DIMN(B)
30 A=SQR(B)
40 Z=INT(A)
50 FORX=3TOZSTEP2
55 FORW=X*3TOBSTEPX*2
60 N(W)=1
70 NEXT
80 NEXT
90 FORA=3TOB-1STEP2
100 IFN(A)=0THENPRINTA;
110 NEXT
112 PRINT
115 PRINT(65535-DEEK
(630))/100"SECONDS":STOP
```

Orics Rule!

By A. M. Brown

```
5 REM**ORICS RULE!**
10 CLS:PAPER0:INK7
20 PRINT:PRINT:PRINT
30 PRINTCHR$(4);CHR$(27);"N
**ORICS RULE!**"
40 FORC=1TO5
50 PRINT:PRINT
60 PRINT"*****"
*****"
70 INK6
80 WAIT50
90 NEXTC
100 PRINTCHR$(4)
110 T$="14321432141414321"
120 FORI=1TOLEN(T$)
130 N=ASC(MID$(T$,I,1))-47
140 PLAY7,0,5,2500
150 MUSIC1,1,N,10
160 MUSIC2,2,N,10
170 MUSIC3,3,N,10
180 WAIT7:NEXT
190 PLAY0,0,0,0
200 WAIT50:EXPLODE
210 CLS:PAPER4:INK6
220 WAIT20:PAPER1:INK3
230 WAIT20:PAPER0:INK7
240 PRINT:PRINT:PRINT:PRINT:
PRINT:PRINT:PRINT:PRINT:
PRINT:PRINT:PRINT:PRINT
250 PRINT:PRINT"
*":WAIT100:INK0:CLS
```

Pattern Recognition

By G. M. Phillips

```
5 REM BY GEOFF PHILLIPS OCT '83
10 HIMEM#17FF: DIM A%(100): POKE#26A, 11
15 GOSUB2000
20 HS=0: GOSUB7000
30 PRINT: PRINT: PRINT "Enter starting level (1-95)": INPUTS$: S=VAL(S$)
: IFS>95ORS<1 THEN30
35 SC=0: L=S: GOSUB5000
40 PRINT: PRINT: PRINT "score: "SC" level "L
50 PRINT: PRINT "Press a key to start": GETZ$
70 GOSUB9000
80 PRINT: PRINT: PRINT "Use ARROWS to enter pattern"
90 FORI=1TOL
92 IFKEY$<>" " THEN92
95 GETZ$: IFZ$="" THEN95
100 Z=ASC(Z$): IFZ<80RZ>11 THEN95
101 GOSUB8000
102 IFZ=A%(I) THENGOSUB6000: NEXT: GOSUB8000: GOTO150
105 PRINT: PRINT: PRINT "W R O N G !!!"
107 PING
110 FORI=10T0300STEP2: PLAY1, 0, 0, 0: SOUND1, I, 10: NEXT: PLAY0, 0, 0, 0
115 WAIT200: PRINT: PRINT: PRINT "This is what the sequence was:": WAIT6
0: GOSUB9000
120 PRINT: PRINT: PRINT "Your score was "SC"
130 IFSC>HSTHENHS=SC
140 PRINT: PRINT "high score: "HS" Please press a key.": GETZ$: GOTO30
150 SC=SC+L*10: L=L+1: IFL<101 THEN40
160 TEXT: PRINT: PRINT "CHEAT!!": EXPLODE
170 END
2000 CLS: PRINT "Pattern Recognition - Geoff Phillips": PRINT
2010 PRINT "In this game you will be shown a ": PRINT
2020 PRINT "series of coloured circles, with a ": PRINT
2030 PRINT "musical tone, which will appear in": PRINT
2040 PRINT "the top, bottom, left or right of": PRINT
2050 PRINT "the screen.": PRINT
2060 PRINT "You must then re-enter that pattern": PRINT
2070 PRINT "by using the arrow keys, corresponding "
2080 PRINT "to the position of the circles that": PRINT
2090 PRINT "you saw. The number of circles will": PRINT
2100 PRINT "be increased when you are correct, and "
2110 PRINT "your score will be increased.": PRINT
2140 PRINT " CHR$(27)"Apress any key to begin"
2145 REPEAT: A=RND(9): UNTILKEY$<>" "
2150 RETURN
5000 REM GENERATE ARRAY OF NUMBERS
5010 FORI=1T0100: A%(I)=RND(9)*4+8: NEXT
5020 RETURN
6000 REM PLAY NOTE AND SHOW CIRCLE
6010 PLAY7, 0, 1, 500: K=(A%(I)-6)*2: MUSIC1, 2, K, 0: MUSIC2, 3, K, 0: MUSIC3, 4
, K, 0
6020 ONA%(I)-7GOTO6040, 6050, 6060, 6070
6040 CURSET0, 67, 3: FILL67, 1, 4: GOTO6080
6050 CURSET120, 67, 3: FILL67, 1, 3: GOTO6080
6060 CURSET0, 133, 3: FILL67, 1, 7: GOTO6080
6070 CURSET0, 0, 3: FILL67, 1, 1
6080 RETURN
6090 RETURN
6500 WAIT50-L/3: RETURN
7000 REM SETUP SCREEN
7010 HIRES
7015 PRINT "Please hang on a tic..."
7020 GOSUB8000
7030 CURSET120, 33, 1: GOSUB7500: CURSET40, 100, 3: GOSUB7500
7040 CURSET200, 100, 3: GOSUB7500: CURSET120, 167, 3: GOSUB7500
7050 RETURN
7500 REM FILL IN CIRCLE
7510 FORI=1T030: CIRCLEI, 1: NEXT: RETURN
8000 REM WIPE ALL CIRCLES
8010 CURSET0, 0, 3: FILL200, 1, 0: CURSET120, 67, 3: FILL67, 1, 0
8020 RETURN
9000 FORI=1TOL: GOSUB6000: GOSUB6500: GOSUB8000: NEXT: RETURN
```

Oric Quickies

Etch-a-sketch

By S. Thomas

The game is based on the game Etch a sketch, it is in HIRES mode and the object is to draw a picture with the dot that starts off at curset 120, 100. The commands are quite simple

U=Up

D=Down

L=Left

R=Right

C=Circle 50,1

Z=Circle 10,1

X=Stop

```
0 HIRES
1 REM***ETCH-A-SKETCH***
2 LETA=120
3 LETB=100
4 CURSET A,B,1
5 GETA$
6 IFA$="U"THENLETB=B-1
7 IFA$="D"THENLETB=B+1
8 IFA$="L"THENLETA=A-1
9 IFA$="R"THENLETA=A+1
10 IFA$="C"THENCIRCLE50,1
11 IFA$="Z"THENCIRCLE10,1
12 IFA$="X"THENSTOP
13 GOTO4
```

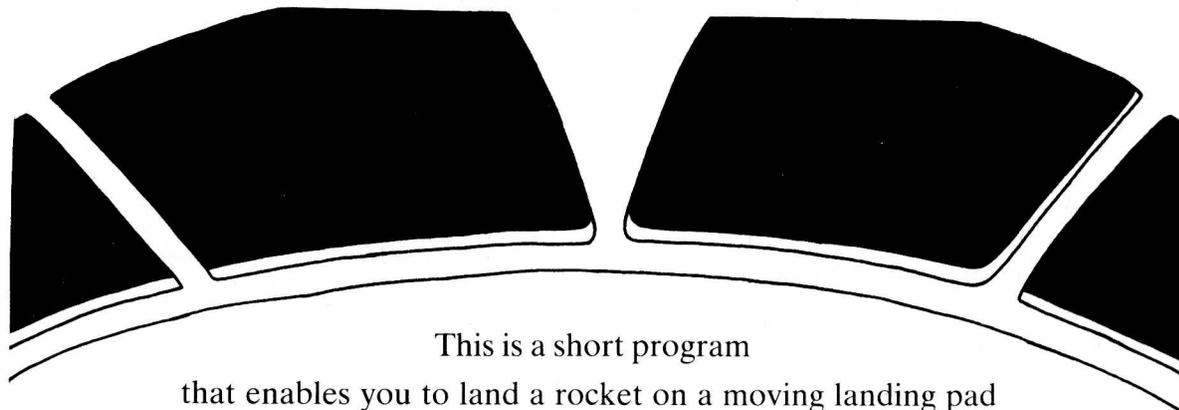
"Play Your Cards Right"

By M. Shaer and S. Singer

```
5 REM**BY M.SHAER&S.SINGER**
10 CLS
20 PRINT"THIS IS A CARD GAME BASED ON THE T.V. PROGRAM PLAY YOUR CARDS RIGHT."
30 PRINT"You have to say what the next suite is going to be."
40 PRINT"1=CLUBS"
45 PRINT"2=HEARTS"
50 PRINT"3=DIAMONDS"
55 PRINT"4=SPADES"
90 S=INT(RND(1)*4)+1
100 INPUTA
110 IFA<>STHEN200
130 IFA=STHEN250
200 PRINT"Sorry you are wrong"
210 GOSUB260
250 PRINT"Well done you are right"
255 W=W+1
260 PRINT"Do you want another go (Y/N)"
265 INPUTA$
270 IFA$="Y"THEN10
280 IFA$="N"THEN300
300 CLS:PRINT:PRINT:PRINT:PRINT" YOUR TOTAL WAS:";W"RIGHT"
```

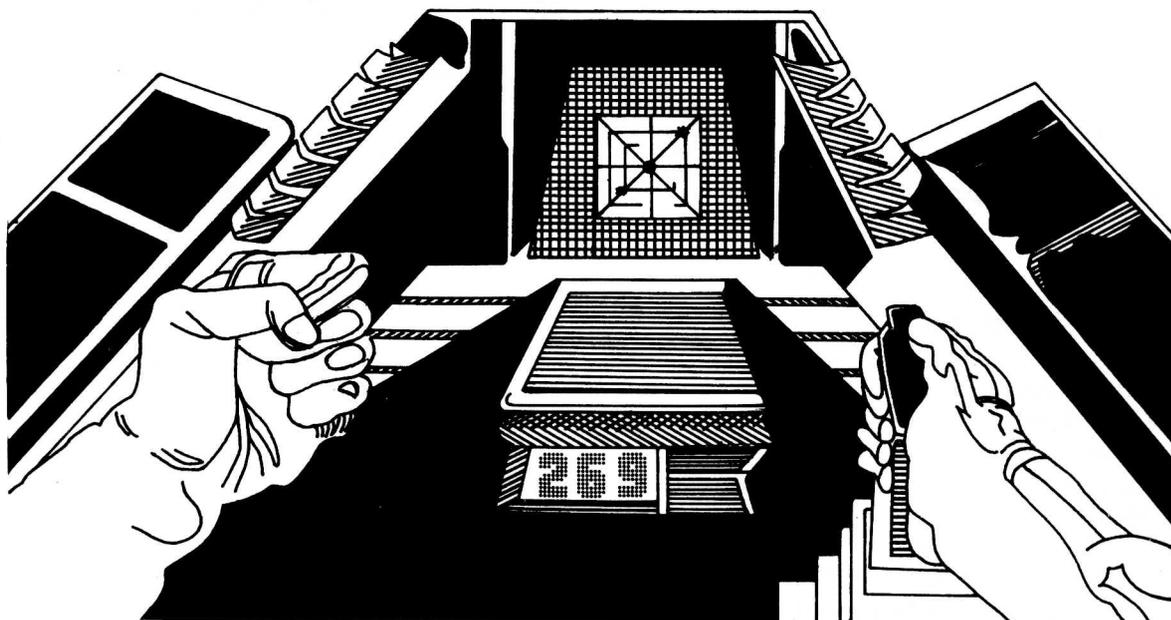
Oric Quickies

'Lander' designed by Jason L. Hall



This is a short program
that enables you to land a rocket on a moving landing pad

```
10 CLS
20 PRINT"XXXXXXXXXXXXX LANDER XXXXXXXXXXXXXXXXX"
30 PRINT" THE OBJECT OF THE GAME IS TO LAND
40 PRINT" THE ROCKET ON THE LANDING PAD USING
50 PRINT" THE LEFT AND RIGHT CURSOR KEYS."
60 PRINT"          GOOD LUCK!"
70 PRINT"PRESS ANY KEY TO CONTILOGUE"
80 GETCS
90 POKE#26A,10
100 CLS
110 XX%=18:YY%=5
120 POKE49040,17:POKE49080,17
130 DI=INT(2*RND(1))+1
140 X=INT(30*RND(1))+1
150 T=200
160 PLOTX,24,127:PLOTX+1,24,127:PLOTX+2,24,127
170 PLOTXX%,YY%,65
180 PLAY7,0,0,0
190 REPEAT
200 PLOTX,24," "
210 IFDI=LANDX>=30THENDI=2
220 IFDI=2ANDX<=2THENDI=1
230 IFDI=1THENX=X+1
240 IFDI=2THENX=X-1
250 PLOTX,24,127:PLOTX+1,24,127:PLOTX+2,24,127
260 PLOTXX%,YY%,32
270 T=T+30:SOUND1,T,10
280 K$=KEY$:IFK$=""THENK$=L$
290 IFK$=CHR$(8)THENXX%=XX%-1:L$=K$
300 IFK$=CHR$(9)THENXX%=XX%+1:L$=K$
310 YY%=YY%+1
320 IFXX%<=1THENXX%=1:IFXX%>=38THENXX%=38
330 IFSCRN(XX%,YY%)=127THEN420
340 PLOTXX%,YY%,65
350 WAIT20
360 UNTILYY%=24
370 CLS
380 EXPLODE
390 PRINT"YOU'VE CRASH LANDED!"
400 WAIT400
410 GOTO460
420 PRINT"PERFECT LANDING!"
430 PING
440 PLOTXX%,YY%-1,65
450 WAIT400
460 CLS
470 PRINT"DO YOU WANT ANOTHER GAME (Y/N)"
480 REPEAT
490 GETA$
500 UNTILA$="Y"ORAS="N"
510 IFA$="N"THENGOTO560
520 CLS
530 PRINT"DO YOU REQUIRE INSTRUCTIONS? (Y/N)"
540 REPEAT:GETZ$:UNTILZ$="Y"ORZ$="N"
550 IFZ$="Y"THENGOTO10ELSE100
560 PAPER0:EXPLODE
570 END
```



Oric Quickies

The following Quickie "Greatest Common Denominator" uses Euclid's G.C.D. algorithm to find the greatest common denominator of two numbers, A and B. A1 and B1 are assigned the values of A and B respectively so that the original numbers in A and B will be preserved.

There is no G.C.D. if either of the numbers is negative. If one of the numbers is zero then the G.C.D. is just the other number (which may also be zero).

A1 must be greater than B1. If it is not line 50 swaps their values so that it is. Line 60 uses recursion to calculate the G.C.D., by calling itself until B1 equals zero. When this occurs A1 contains the G.C.D. of the two numbers and the value is printed out.

"Greatest Common Denominator"

By Gary Nugent

```
1 REM ** Greatest Common Denominator **
2 REM ** (C) Copyright G.Nugent, 1983 **
10 INPUT "Please enter 2 numbers: ";A,B
20 A1=A:B1=B
30 IF A1<0 OR B1<0 THEN PRINT "No Greatest
Common Denominator.":GOTO 80
40 IF B1=0 THEN 70
50 IF A1<B1 THEN T=A1:A1=B1:B1=T
60 IF B1<>0 THEN T=A1:A1=B1:B1=T-INT(T/B1)
*B1:GOTO 60
70 PRINT "GCD = ";A1
80 END
```

CARD SHUFFLE

By Gary Nugent

```
5 REM ** CARD SHUFFLE **
10 REM (C) COPYRIGHT G.NUGENT, 1983
20 DATA AH,2H,3H,4H,5H,6H,7H,8H,9H,TH,JH,QH,KH
30 DATA AS,2S,3S,4S,5S,6S,7S,8S,9S,TS,JS,QS,KS
40 DATA AD,2D,3D,4D,5D,6D,7D,8D,9D,TD,JD,QD,KD
50 DATA AC,2C,3C,4C,5C,6C,7C,8C,9C,TC,JC,QC,KC
100 DIM CD$(51)
105 REM ** Read cards into array **
110 FOR I=0 TO 51:READ CD$(I):NEXT I
115 REM ** Shuffle cards **
120 FOR I=0 TO 51
130 CD=INT(RND(1)*52)
140 T#=CD$(I):CD$(I)=CD$(CD):CD$(CD)=T#
150 NEXT I
```

The following Quickie "Card Shuffle" will be useful to those intending to write their own card-game programs. The routine shuffles suited cards which are required in Poker or Pontoon, for example.

The letters standing for the suits (H-hearts, S-Spades, D-diamonds, C-clubs) can be redefined to be the actual suit characters. T, J, Q and K stand for Ten, Jack, Queen and King respectively. The "T" can be redefined to be a "10" symbol if you like.

The only problem with the program is the use of RND which produces the same sequence of "random numbers" each time the computer is switched on. This can be cured by using the method of creating a random seed described in the last issue of "ORIC OWNER".



Totally devoted to you!

Oric Owner is the official magazine devoted to the Oric 1 and Atmos home computers.

It's crammed full of in-depth information, advance news on the latest add-ons, superb programs and interviews with the engineers who designed it.

The first issue is absolutely free when you buy your Oric, so why not keep ahead of the latest developments and subscribe to further issues. A years subscription of 6 issues is now only £10 (£15 overseas) so post the coupon today.

Can you imagine life without it?



ORIC OWNER SUBSCRIPTION FORM

Please send me the next 6 issues

I enclose a cheque for (cheques payable to Tansoft Ltd)

Name.....

Address.....

..... Postcode.....

Back issues are available for £1.20 each. If you missed your first free issue contact your dealer or Oric Products International who will supply you with one.

TANSOFT Tansoft Ltd, Units 1 & 2
Cambridge Techno Park Newmarket Road, Cambridge CB5 8PB

TANSOFT are pleased to announce the prices for the Oric printer accessories which are as follows:

Set of 4 colour pens	£3.99
	(per set)
Printer rolls	£1.99
	(per roll)

Above prices are inclusive of V.A.T. and P.+P. and are available from stock. Trade enquiries welcome.

Stop Press

Tansoft will shortly be releasing three new packages for the Oric and Atmos. These are Oric-Mac, a macro-assembler, Pascal and a Basic Compiler. These are being written in cooperation with Oasis Software.

ALL: TRONICS,
297 BRUS. STEENWEG,
1950 KRAAINEM
Tel: (09)(32)(0)2 7678223

Authorized Benelux Distributor

ALL: TRONICS

ORIC 48K
ORIC 16K
ORIC Games Software
ORIC Assembler and
Professional Software

Joysticks
Printers
Manuals (*Dutch & French*)
Extension boxes
A/D converter
I/D ports
Heavier Power Supply,
Ram upgrades etc.

ORIC-1 SOFTWARE COMPARE THESE PRICES

XENON-1	IJK	£7.50
INVADERS	IJK	£6.75
REVERSE	IJK	£5.95
FANTASY QUEST	IJK	£5.95
3D MAZE & BREAKOUT	IJK	£6.75
CANDYFLOSS & HANGMAN	IJK	£6.75
ZODIAC	A&F	£6.20
DEATH SATELLITE	A&F	£6.20
THE ULTRA	PSS	£6.40
LIGHT CYCLE	PSS	£6.40
CENTIPEDE	PSS	£6.40
THE GAUNTLET	PSS	£6.40
INVADERS	PSS	£6.40
HOPPER	PSS	£6.40
ORICMON	PSS	£7.95
FORTH	TANSOFT	£13.50
ORIC BASE	TANSOFT	£13.00
CHESS	TANSOFT	£8.95
ORIC MUNCH	TANSOFT	£7.00
RAT SPLAT	TANSOFT	£7.00
DEFENCE FORCE	TANSOFT	£7.00
ORIC CALC	TANSOFT	£13.00
FLIGHT	TANSOFT	£4.00
THE HOBBIT	TANSOFT	£13.50
AUTHOR	TANSOFT	£13.00

ALL PRICES INCLUDE VAT & P&P - ALLOW 21 DAYS FOR DELIVERY
PLEASE SUPPLY

TOTAL ENCLOSED £

NAME

ADDRESS

SEND ORDERS TO:- **POSTAL COMPUTER SERVICES,**
FREEPOST P.O. BOX 2, MIDHURST, WEST SUSSEX, GU29 9BR.

Dormere Software Now Presents Backgammon

(Reviewed by Tansoft
in this issue)

A superb version of the classic game
of backgammon for your 48k Oric.

features:-

- 3 modes of play
- Machine code routines
- Hi-Res Graphics
- Coloured board
- Sound effects

Excellent
value at
£7.95



Orical Invaders

A super-fast version of the ever
popular space invaders game.

features:-

- Hi-Res Graphics
- Full sound effects
- Hall of fame
- Written in machine code
- Fast/Slow bombs etc.
- Bonus Bases
- 16k or 48k Oric

Only
£5.95

and if you haven't had enough yet.
Try the ultimate in mind bending
games for all the family, **16/48k
Orical Games Pack 1.**
Stretch your intelligence with
Symon, Brain strain, OXO and
Battleships for only **£4.95.**

Dealer's and Distributor's enquiries
welcomed.

Cheques and P.O.s made payable
to:- **Dormere Software Limited,**
Belgreen House, Green St.,
Macclesfield, Cheshire.
Tel: Macclesfield 619535.

PSS, creating programs for your ORIC 1....

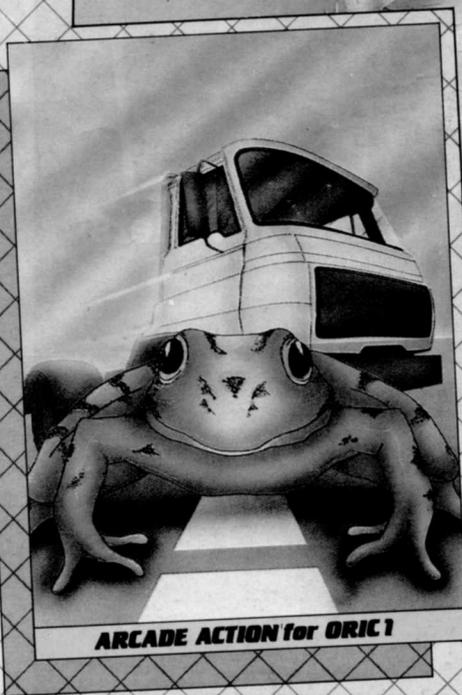


Moon- base Alpha

A race against time, fast exciting arcade action.
£6.95

The Ultra

The ultimate challenge, constant waves of alien mutants, fast action.
£6.95



ARCADE ACTION for ORIC 1



Hopper

Can you help Fergy and his friends get across the 4-lane high way and back to the lilly pond?

£6.95



PSS SOFTWARE

FOR INSTANT CREDIT CARD SALES TEL (0203)667556, SEND CHEQUE OR P.O.
TO P.S.S. 452 STONEY STANTON RD. COVENTRY CV6 5DG.
TRADE ENQUIRIES CONTACT JOHN FLETCHER, COVENTRY (0203) 81346