The U.K. ATARI Computer Owners Club Issue 8 Price £1.00

Independent User Group

Monitor

Moonbase Plato – Adventure at its Best

New Atari Computers Previewed

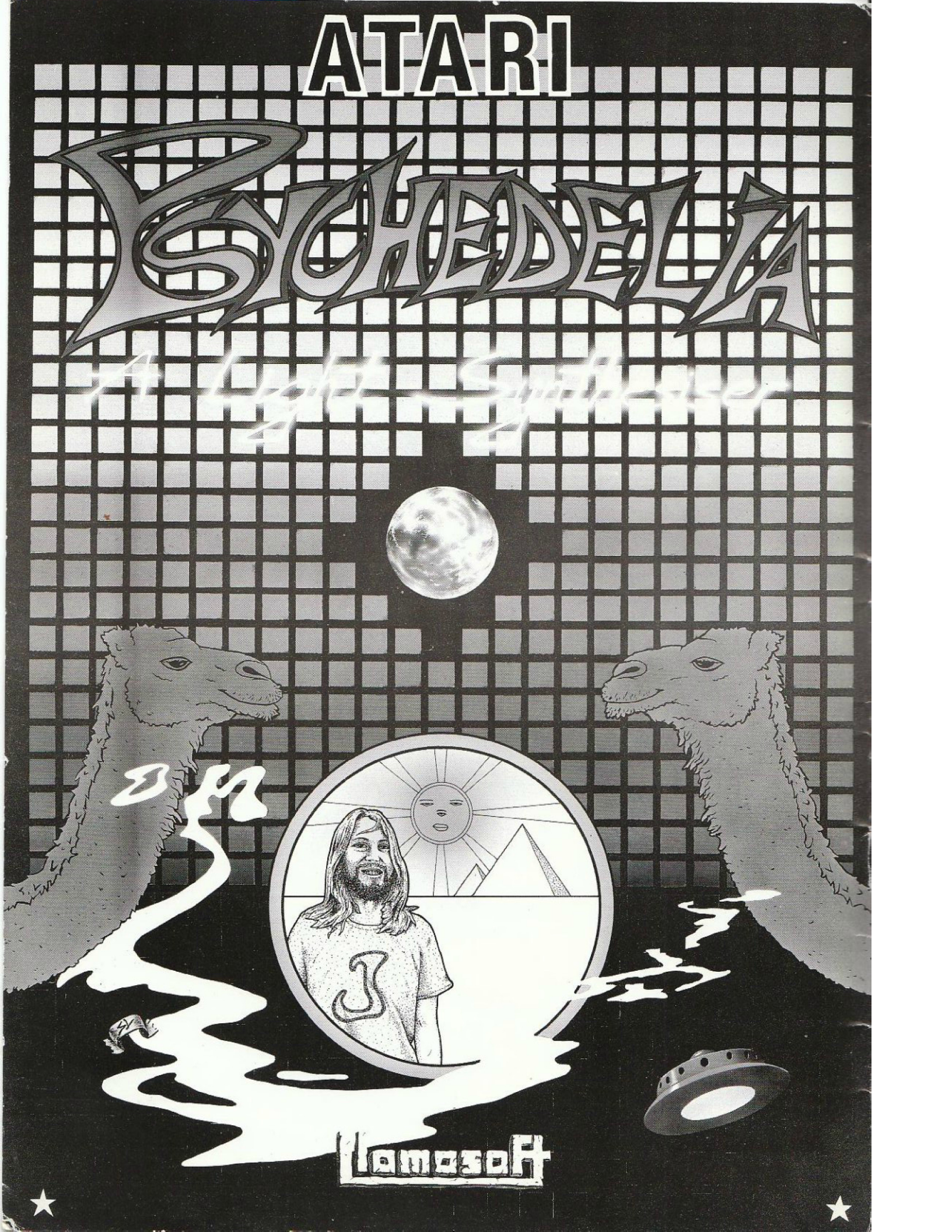
Horizontal and Vertical Scrolling Techniques

Nightmare Reflections – Excellent Miniadventure

ATARI

PSYCHEDELIA

A Light Simulator



Llamasoft

THE U.K.

ATARI

COMPUTER OWNERS CLUB

INDEPENDENT USER GROUP

NEW ERA?

You must be deaf and blind if you have not heard about the new Atari computers soon to be released on an awe-struck world. From all reports the new machines are stunning in both capability and price. I am sure many of you already have plans to upgrade to the 'Jackintosh' at the earliest opportunity, I know I am! I just hope that Jack can deliver the goods when he says he will, if he does then I think Atari will rise to the top once again and a new era will have dawned.

There seems to have been a general upsurge in Atari sales over the christmas period obviously due to the low prices of 800XL's, and now with the even lower 800XL packs at £129.95 for 800XL, 1010, and Pole Position and Intro to Programming tapes, or £249.95 for 800XL, 1050, Home Filing Manager and Payoff (new adventure) disks, sales could go even higher. If you have wanted to get a Disk Drive, now is the time to do it!

Other good news for Atari fans is that a new magazine will soon be available in the book shops. It is being published by Database Publications who currently print mags for the Apple and Amstrad and it will be called Atari User. It will be monthly and retail for £1.00, it is to be launched at the Hanover Computer Fair in mid-April along side the European launch of the new Atari micro's. Database are offering £3.00 off of a £12.00 yearly subscription to club members, so send a cheque made payable to Database Publications for £9.00 to us here at HQ and we will pass it on to Database. Note this offer is valid until 30th June 1985.

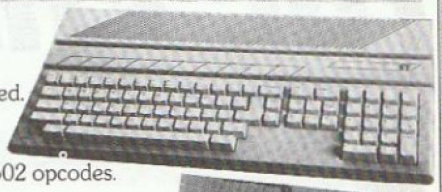
Everything is not all roses however, we are sad to report the demise of Channel 8 who brought you the fabulous Mysterious Adventure series, these superb adventures are not lost however, as they have been sold to Adventure International/Calisto, so they should still be available.

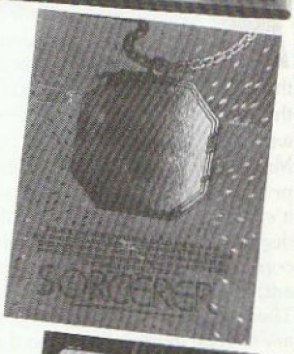
On the home front you have probably noticed that the club magazine has been given a name. We did this because it is being sold in a lot more outlets than ever before and we thought that going into a shop and asking for "the magazine of the U.K. Atari Computer Owners Club" was quite a mouthful. The name Monitor was picked from a short list of five names, and we felt that it had dual points in its favour as it not only had computer conitations, i.e. TV type monitor, but also means keeping track of Atari events. We actually wanted to call it "Atari Monitor" but were politely told by Atari (U.K.) that we could not! Oh well, it was to be expected I suppose.

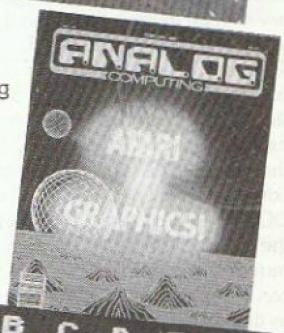
CREDITS

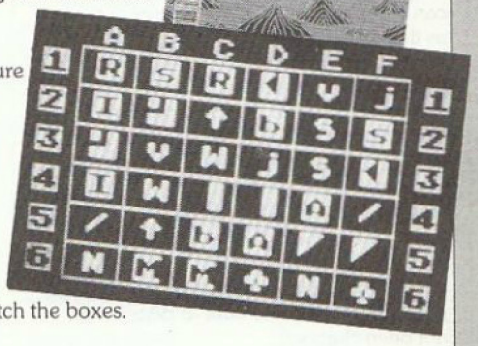
Editor	Roy Smith
Art Editor	Peter Blackmore
Technical Editor	Ron Levy
Technical Editor	Keith Mayhew
Adventure Editor	Steve Hillen

CONTENTS









SUBSCRIPTION RATES

U.K. and Ireland	£4.00
Europe	£7.00
Outside Europe (Surface)	£7.00
Outside Europe (Airmail)	£10.50

The above prices are in English Pounds Sterling and include postage and packing.

A subscription/membership fee to join the U.K. Atari Computer Owners Club is just £4.00 for four issues of the club magazine. All cheques/postal orders are to be made payable to the 'U.K. Atari Computer Owners Club'. Overseas membership is also available at slightly higher rates. Overseas members who use the Library service should include enough extra monies to cover return postage.

ADVERTISEMENTS

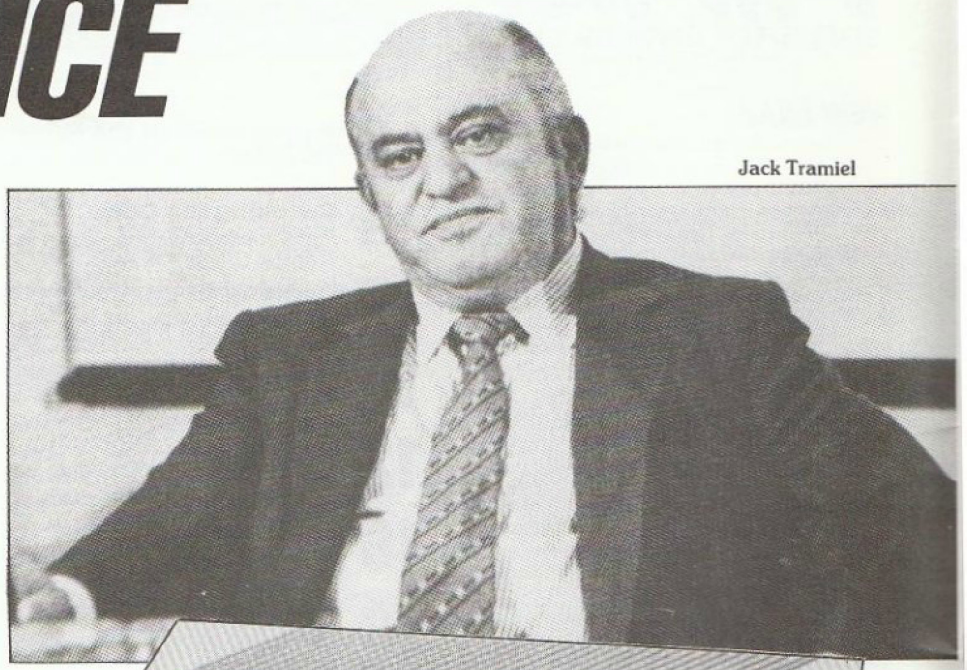
Please note that the club cannot be held legally responsible for claims made by advertisers.

COVER: 'Monitor' and 'Moonbase Plato' are trademarks and are copyright of the U.K. Atari Computer Owners Club.
CLUB ADDRESS: The U.K. Atari Computer Owners Club, P.O. Box 3, Rayleigh, Essex.

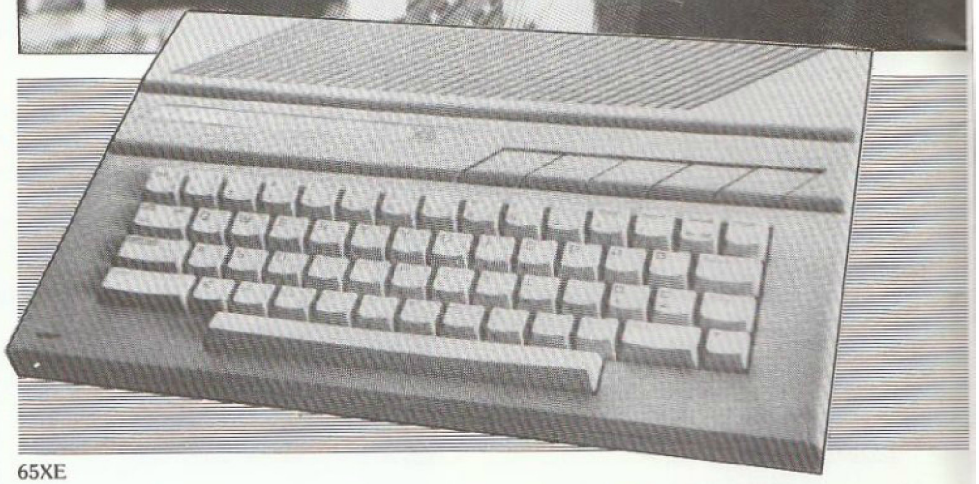
Copyright: "The UK ATARI COMPUTER OWNERS CLUB" is an independent users group and is in no way affiliated with ATARI. All material is subject to world wide Copyright protection, and reproduction or imitation in whole or part is expressly forbidden. All reasonable care is taken to ensure accuracy in preparation of the magazine but the UK ATARI COMPUTER OWNERS CLUB cannot be held legally responsible for its contents. Where errors occur, corrections will be published as soon as possible afterwards. Permission to reproduce articles or listings must be sought from the UK ATARI COMPUTER OWNERS CLUB. ATARI (and any other Atari product mentioned in the magazine) is a trademark of ATARI CORPORATION.

POWER WITHOUT THE PRICE

by Eddie Taw



Jack Tramiel



65XE

Atari's new slogan, "Power without the Price", was daubed everywhere on their display stand at the Las Vegas Consumer Electronics Show at the beginning of January, and if you think that's a big claim to make then don't worry, because it's really true. It looks as if Mr. Tramiel has really done what he promised, and is all set to restore Atari to it's rightful place at the top of the tree. On display were the two new ranges of Atari computers, the XE (6502 based XL like) and the ST (68000 based Macintosh like). There are four in the XE range and they are all 8 bit micro's based on the 6502 and as such are claimed to be XL compatible. The basic model is the 65XE which is fundamentally an improved 800XL with simplified circuitry and combined chip functions to give a more reliable machine than the 800XL whilst maintaining 100% compatibility. The XE line will use a new DOS 2.5 which is said to be very similar to the classic DOS 2.0S. The new DOS is required in order that the new machines can accept the current 1050 drive as well as the new 500K, 3 1/2" disk drives. The new DOS was written by Bill Wilkinson of OSS who developed the old Atari Disk Operating System. The next machine in the range is the 130XE which is the same as the 65XE but carries 128K of memory. The other good point is that it will maintain an open parallel bus for plug-in peripherals. The PBI will even be improved over the current XL format, with improved timing and a built in +/– 5V power amplifier. The built-in BASIC has not been changed.

A portable version of the 65XE will be available, the 65XEP, which comes with a built-in 5" very clear green screen monitor and a 3 1/2" disk drive. The fourth in the range will not be ready until the middle of the year. It is to be a music/synthesiser based machine, the 65XEM, when the polyphonic AMIE sound chip is finished (around March) it is supposed to go into this micro. The AMIE chip is rumoured to be capable of emulating human speech and singing with unprecedented accuracy.

65XEP

The power of these machines must be seen to be believed, but the real breakthrough for Atari will be the prices! The 65XE will be under $120, the 130XE is "well under $200", maybe as low as $150, the 65XEP will be under $400, the price of the 65XEM is to be announced at a later date but is thought to be likely to sell for $160.

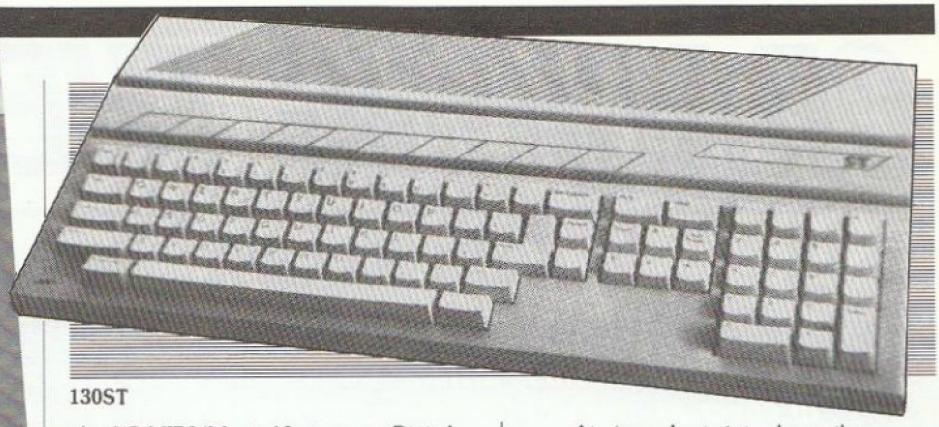
Manufacture of the new computers is not expected to begin until early March, so they may not be available until May or June, but I am sure that with the new dynamic image of the Tramiel's we can see them in the shops as soon as they are ready.

As part of the XE launch demonstration a new program called INFINITY was given it's public debut. It is a truly integrated combination of spreadsheet, word processor and relational database, it can run in 64K (even on the old 800 with only the loss of a few minor features), and costs only $50 (a 16 bit version will cost $70). INFINITY incorporates windows, icons, pull-down menus, integrated printing and a telecommunications package. It also supports multitasking, 2 operations on the 65XE, 3 on the 130XE and 6 on the ST micro's. Matrix Software, who developed INFINITY, claim that they were able to get so much into a 64K memory by two-step "optimizing" of the assembly language compilation, which is a procedure usually only used in advanced military and government software. INFINITY has a truly impressive performance and is more than a match for LOTUS 1-2-3, Framework or Symphony!

Other software shown at CES included Shopkeeper which is an easy to use accounting package for small

businesses, it also keeps inventory and can be used as a cash register. Silent Butler is a personal finance program which balances cheque and credit card accounts and has the unique ability to print your own personalised cheques using a plastic holder that fits into your printer. Also previewed was Music Painter which is a user friendly music constructor that replaces standard musical notation with easy to understand line patterns and icons, in addition it's 3 instrumental voices can be controlled directly from the joystick.

The launch of the XE range was impressive enough but Atari were not content to let matters rest there! Also on display was the new range of 16 bit micro's, the ST range. There are to be three new models, the 130ST (128K version), the 260ST (256K version) and the 520ST (512K version). All three machines are based on the Motorola 68000 chip which is also used by Apple's Macintosh, I suppose it was inevitable therefore that the new range would be nick-named "Jackintosh's". Each model incorporates a 192K built-in ROM containing Digital Research's GEM (Graphics Environment Manager) and 68K Operating System. They will be capable of running Logo, C and Pascal. The casings have been completely restyled to give an impressive business like appearance, the keyboards include a full selectric-style layout, a 10 key number pad, a cursor pad with Help and Undo keys, and 10 function keys. The rear of the machines is covered in connector ports, including a Centronics port, an RS232 port, interfaces for disk drives, joysticks and mice, plus four video ports: T.V., Composite Video, RGB and Hi-res Monochrome. The CPU features eight 32 bit data registers, eight 32 bit address registers, a 16 bit data bus and a 24 bit address bus. They have 32K bit mapped screens with a choice of three graphics modes, a 320 x 200 pixel 16 colour mode, a 640 x 200 pixel 4 colour mode and a 640 x 400 pixel monochrome mode. The machines have a range of 512 colours with 8 levels of green, red and blue. A special sound chip is included which is capable of giving controllable frequencies in the region of 30Hz to supersonic!! Three channels are available with separate volume and frequency control. There are



130ST

also I/O MIDI (Musical Instrument Digital Interface) ports which were displayed driving the new Casio C2101 Synthesiser.

The new GEM software could be the jewel in the Atari crown. GEM simplifies the users interface with the machine by employing the use of graphical images. Thus keyboard input is avoided and quick information selection is achieved by the use of a joystick or mouse attachment. Part of the GEM package is the AES (Application Environment Services) supplement which includes subroutine libraries to monitor and respond to user input from mouse movement, mouse button clicks or keyboard entry. The user is also allowed to write a menu in text form which can then be translated into pull-down menu form, and there are several storage/retrieval managers to keep an eye on icons, graphics and the screen under a pull-down menu.

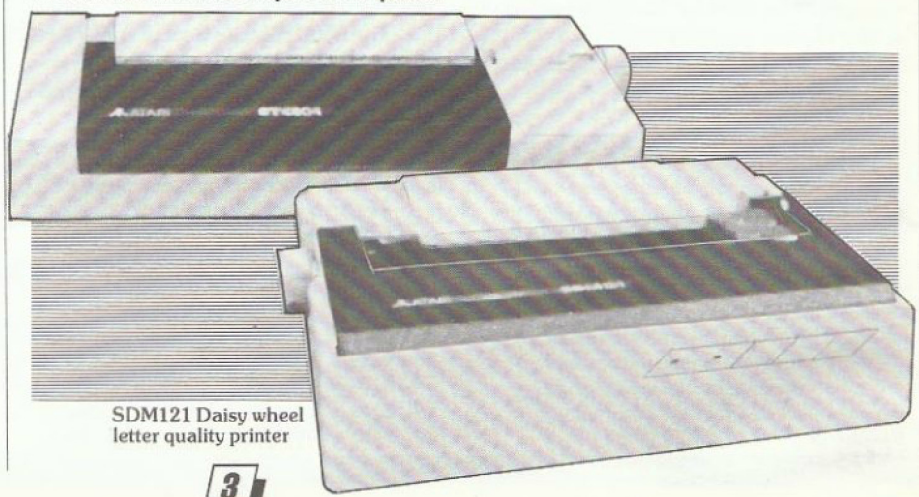
PERIPHERALS

Atari have introduced a new, sleek, compact 3 1/2" disk drive with 500K capacity which is compatible with both the XE and the ST ranges. The new drive, designated SF354 will be under $200 and there is talk of a 250K version (SF324) for around $150. Not shown at the show, but said to be in the pipeline is a 10 Megabyte hard disk drive (SH317) for under $600 and there are rumours of a 15 Megabyte version. The speed of these drives is reported to be 1.3 Megabytes per second for the ST range, and it is hoped to increase from 19,200 to 30,000 for the XE range. For 5 1/4" floppy users a replacement for the 1050 will be introduced in the style of the XE, tentatively coded XF521.

Atari are also to introduce other peripherals in the form of printers, monitors and communications equipment. There are to be 4 printers which will be suitable for both the XE and the ST range. For $150 you will be able to buy a true letter quality (but slow at 12 cps) daisywheel, or an 80 cps dot matrix that produces graphics as sharp as the Apple Imagewriter, or a 50 cps non-impact colour printer that produces very clear text, or for $99 a black only 20 cps printer. Several monitors have been announced and these include the XC141 composite colour 14" model, the XM128 is a 12" green screen monitor with built-in 80 column card mainly for use with software on the XE models. Especially designed for use with the ST range is the SC1224 12" RGB analogue colour monitor which can display 512 colours on screen, and it is said there will also be a version of this monitor with a built-in 3 1/2" disk drive. Finally, the SM124 is an extra high resolution monochrome monitor for use with the ST models. In the telecommunications field a direct-connect 300 baud modem, the XM301, will be available for $50 and it will be supplied with uploading and down-loading software. Obviously because of the different communications laws in this country the XM301 is unlikely to be available here, but there again a European standard modem may not be out of the question, only time will tell.

Well there it is folks, it looks as if Jack really can do it and pull Atari's fat out of the fire. It is quite likely that the new machines won't arrive in the U.K. until mid-summer, but when they do if only half the reports are true then Atari can look forward to rising to the top of the tree where they should always have been!

STC504 Dot matrix non-impact colour printer



SDM121 Daisy wheel letter quality printer

CRACKING THE CODE

by Keith Mayhew Part Four

In the three previous articles on machine code programming we have tried to cover as much of the necessary background needed. Do not worry if you have read the articles over and over again and yet still feel a little lost! Machine code programming only comes with experience and practice makes perfect! Now the remaining 6502 opcodes will be explained and a complete reference chart of all the instructions is shown.

More Logical Operators

The last four logical operators to be covered allow us to move all the bits in one byte simultaneously. The first two instructions are 'ASL' and 'LSR', these mean 'arithmetic shift left' and 'logic shift right'. 'ASL' moves all the bits to the left, therefore bit 0 moves to bit 1, bit 1 moves to bit 2, etc. A zero is always placed in bit 0 afterwards and bit 7 is 'dropped' into the carry flag. In effect 'ASL' multiplies a byte by a factor of two, with the most significant bit in the carry. 'LSR' is the converse of 'ASL', it moves all the bits to the right, so bit 7 moves to bit 6, bit 6 moves to bit 5, etc. Bit 0 is dropped into the carry flag, and a zero is placed into bit 7. This has the effect of dividing the byte by a factor of two, the number would of course be rounded down to the nearest whole number, but the carry flag would indicate a 'half' if it was set, i.e. bit 0 divided by two, or an exact result if the carry flag was clear. These two instructions are also used to test certain bits within a byte, e.g. if you wanted to test the second bit then you would use 'LSR' twice, then the contents of the second bit would be in the carry flag.

The last two logical operators are 'ROL' and 'ROR' which are very similar to the previous two. They mean 'rotate left' and 'rotate right' and perform a 9 bit rotation, in the respective direction, with the carry as the ninth bit, i.e. all the bits are shifted as in the previous instructions except that the previous carry is placed into the bit which would normally have been set to zero. These four operations are illustrated in Figure 1.

These shift operations can be used on any memory location and also the accumulator. If the accumulator is used then this is indicated in the operand by the symbol 'A', e.g. 'ROL A'. This is an exception in the assembly language because the 'A' does not generate an operand byte when assembled, unlike a memory address. It simply indicates to the assembler to generate the op-code which implies use with the accumulator. It is also possible to combine the use of 'ASL' with 'ROL' and 'LSR' with 'ROR' to perform 16 bit, or more, shifts in either direction. As an example try and see what happens when 'ASL' is followed by one (or more) 'ROL's, on sequential memory locations, note how the carry passes one bit between each byte.

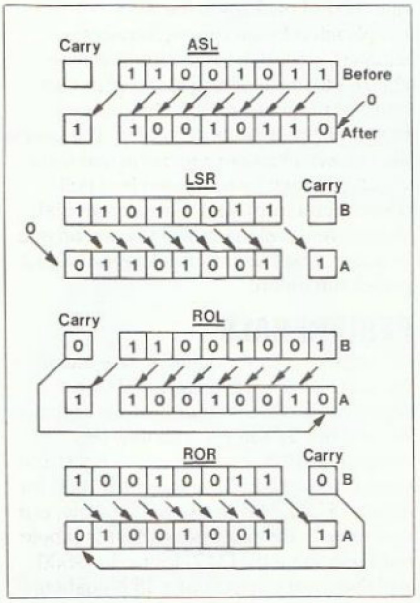


Figure 1.

Decision Making?

The 6502 contains a set of instructions which can alter the program's flow, these are very important and are frequently used to pass control to another part of the program or to repeat a piece of code several times in a loop. These instructions can be either unconditional or conditional.

There are only two instructions in the 6502 which pass control unconditionally, they are jump, 'JMP' and jump to subroutine, 'JSR'. The jump instruction simply places a new address into the program counter, PC, and thus continues program flow from this new address, e.g. JMP $4010 will cause the instructions starting at $4010 to be executed. The other instruction, JSR, allows a subroutine to be called, (refer to Part 2, The Hardware Stack), this causes the return address to be placed on the stack which points to the instruction after the JSR instruction. Control is then passed to the subroutine by placing the subroutine address into the PC register, e.g. JSR $2200 will place the PC's contents plus two on the stack, i.e. the address of the instruction after the JSR $2200. Then the new address is placed in the PC, thus program control is passed to the subroutine. A subroutine is expected to return to the main program at the return address stored on the stack, this is achieved by the 'RTS', return from subroutine instruction. This takes the return address off the stack and places it in the PC, continuing from after the JSR instruction in the main program.

The conditional types execute a piece of code depending on certain conditions. All the 6502's conditional instructions are branches, that is they do not specify an absolute, i.e. fixed address, but give an offset from the current place in the program. The disadvantage of this is that branches have a limited number of bytes backwards and forwards in which they can pass control, this space is one page in size, as specified by a single byte. The advantage of using branches is that if the code is moved to another area of memory it will still execute correctly because the branches are relative to where the program is. The unconditional instructions, JMP and JSR, rely on having the code at a fixed place in the memory, due to their absolute addressing techniques, and therefore code has to be kept at a fixed address when running with these instructions. There are four pairs of branch instructions, each pair being the opposite of each other. Each pair operate depending on the state of a flag in the status register, P. The first pair is BEQ and BNE, these stand for branch if equal and branch if not equal. Each branch depends on the state of the 'Z' flag, BEQ will branch if Z is set and BNE will branch if Z is clear, if a branch fails then control is passed to the next instruction. The state of the Z flag will be dependent on the last instruction to change it, (refer to Part 2,

The Processor's Flags). The rest of the branch instructions are: BPL, branch if plus (including zero) and BMI, branch if minus, (N flag). BCS, branch if carry set and BCC, branch if carry clear, (C flag). BVS, branch if overflow set and BVC, branch if overflow clear (V flag).

The operand byte which follows every branch instruction contains the offset from the first instruction following the branch, i.e. two bytes on from the branch opcode. To allow branching in both directions, the offset is signed using the two's compliment method. The offset is added to the current value of the PC register, which is pointing to the next instruction. This allows a branch of upto 127 bytes on or upto 128 bytes backwards, see Figure 2.

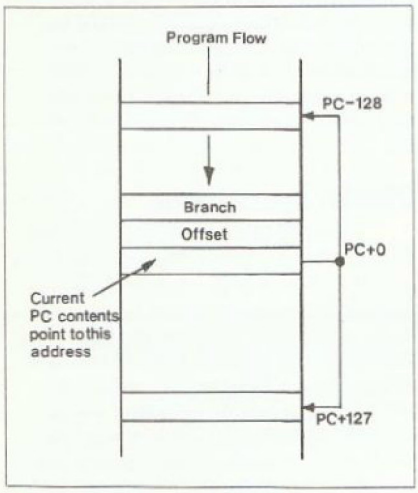


Figure 2.

An easy method for calculating the two's compliment for the offset is the following: If the range lies between 0 and 127 then the number is already in the correct form. If the number is between -1 and -128 then subtract the magnitude of the offset from 256 to get the two's compliment representation, e.g. to get a branch of 100 bytes backwards, i.e. to represent -100, take 100 from 256 to get the correct value of 156.

There exists a method for effectively extending the range of a branch, that is to use a JMP instruction in combination with a branch instruction. For example, if you wanted to branch to a piece of code if the Z flag was set, you would use the BEQ instruction, however, if the code is out of the offset range from that instruction then you can use the following method: Use the opposite branch instruction, BNE in this case, to branch around a JMP instruction, then if the Z flag was set the BNE would fail and pass control to the JMP which would be able to continue from anywhere in memory. Branches can also be used unconditionally if a flag is set to a known state before the branch. We have already covered the instructions to do this which are CLC and SEC (clear and set the carry) which are used with BCC and BCS (branch if carry is clear or set). The last control instruction is CLV which clears the overflow flag, note that there is no instruction to set the overflow flag, this is used with BVC and BVS (branch if overflow is clear or set).

TRANSFER OPERATIONS

	Implied	Immediate	Accumulator	Zero-Page	Absolute	Zero-Page,X	Zero-Page,Y	Absolute,X	Absolute,Y	(Indirect)	(Indirect,X)	(Indirect),Y	Relative	Flags
LDA		A9		A5	AD	B5		BD	B9		A1	B1		NZ
LDX		A2		A6	AE		B6		BE					NZ
LDY		A0		A4	AC	B4		BC						NZ
PHA	48													
PHP	08													
PLA	68													NZ
PLP	28													NVBDIZC
STA				85	8D	95		9D	99		81	91		
STX				86	8E		96							
STY				84	8C	94								
TAX	AA													NZ
TAY	A8													NZ
TSX	BA													NZ
TXA	8A													NZ
TXS	9A													
TYA	98													NZ

ARITHMETIC OPERATIONS

	Implied	Immediate	Accumulator	Zero-Page	Absolute	Zero-Page,X	Zero-Page,Y	Absolute,X	Absolute,Y	(Indirect)	(Indirect,X)	(Indirect),Y	Relative	Flags
ADC		69		65	6D	75		7D	79		61	71		NVZC
DEC				C6	CE	D6		DE						NZ
DEX	CA													NZ
DEY	88													NZ
INC				E6	EE	F6		FE						NZ
INX	E8													NZ
INY	C8													NZ
SBC		E9		E5	ED	F5		FD			E1	F1		NVZC

LOGIC OPERATIONS

	Implied	Immediate	Accumulator	Zero-Page	Absolute	Zero-Page,X	Zero-Page,Y	Absolute,X	Absolute,Y	(Indirect)	(Indirect,X)	(Indirect),Y	Relative	Flags
AND		29		25	2D	35		3D	39		21	31		NZ
ASL			0A	06	0E	16		1E						NZC
EOR		49		45	4D	55		5D	59		41	51		NZ
LSR			4A	46	4E	56		5E						(N=0)ZC
ORA		09		05	0D	15		1D	19		01	11		NZ
ROL			2A	26	2E	36		3E						NZC
ROR			6A	66	6E	76		7E						ZC

CONTROL OPERATIONS

	Implied	Immediate	Accumulator	Zero-Page	Absolute	Zero-Page,X	Zero-Page,Y	Absolute,X	Absolute,Y	(Indirect)	(Indirect,X)	(Indirect),Y	Relative	Flags
CLC	18													(C=0)
CLD	D8													(D=0)
CLI	58													(I=0)
CLV	B8													(V=0)
NOP	EA													
SEC	38													(C=1)
SED	F8													(D=1)
SEI	78													(I=1)

Table 1.

COMPARISON OPERATIONS

	Implied	Immediate	Accumulator	Zero-Page	Absolute	Zero-Page,X	Zero-Page,Y	Absolute,X	Absolute,Y	(Indirect)	(Indirect,X)	(Indirect),Y	Relative	Flags
BIT				24	2C									(N=*B7)(V=*B6)Z
CMP		C9		C5	CD	D5		DD	D9		C1	D1		NZC
CPX		E0		E4	EC									NZC
CPY		C0		C4	CC									NZC

*B7 = BIT 7 OF MEMORY TESTED *B6 = BIT 6 OF MEMORY TESTED

BRANCH OPERATIONS

	Implied	Immediate	Accumulator	Zero-Page	Absolute	Zero-Page,X	Zero-Page,Y	Absolute,X	Absolute,Y	(Indirect)	(Indirect,X)	(Indirect),Y	Relative	Flags
BCC													90	
BCS													B0	
BEQ													F0	
BMI													30	
BNE													D0	
BPL													10	
BRK	00													(B=1)(I=1)
BVC													50	
BVS													70	
JMP					4C					6C				
JSR					20									
RTI	40													NVBDIZC
RTS	60													

Table 1.

Comparison tests are used in conjunction with branches to extend their testing ability on data, the 6502 has four of these comparison instructions. The first is compare, CMP, which subtracts the operand of the compare from the accumulator without storing the result in the accumulator, instead only the Z, N and C flags are changed to indicate the result. The N flag is set if the result was negative (in two's compliment form). There are three possibilties when using this instruction which affect Z and C, they are:

1. If the accumulator is greater than the data then the result will be non-zero and the carry will be set, i.e. Z=0 & C=1.
2. If the accumulator is equal to the data then the result will be zero and the carry will be set, i.e. Z=1 and C=1.
3. If the accumulator is less than the data then the result will be non-zero and the carry will be cleared, i.e. Z=0 & C=0.

There are two other instructions, CPX and CPY which perform the same operation as CMP on the X and Y registers respectively and are used in the same way. The last instruction, BIT, is useful in only a few applications. It tests a memory location against the accumulator by ANDing the two together, in addition it places bit 7 of the memory into the N flag and bit 6 of the memory into the V flag, without changing the accumulator. Thus it makes it easy to test the two top bits by using the appropriate branch instructions. To test any other bit of the memory a '1' should be placed in the corresponding bit of the accumulator, if that bit is also set in the memory then the result will be non-zero and the Z flag will be cleared to indicate this.

Stack Operations

There are two pairs of instructions for this purpose, each pair either pushes a byte onto the stack or pulls a byte off of the stack. PHA pushes a copy of the accumulator onto the stack and PLA pulls the top byte of the stack into the accumulator. PHP and PLP perform the same operation on the P register which contains the status flags. To save or load the X and Y registers it is necessary to use one of the transfer instructions to get the information between the index registers and the accumulator which can then be pushed or pulled.

Interrupt Instructions

There are four instructions which are used with interrupts, two set and clear the interrupt (I) flag, they are SEI and CLI. RTI returns from an interrupt and BRK causes a 'software' interrupt. These instructions will be covered in a future article!

There are two tables in this article which are intended for reference purposes, the first is Table 1 which lists all of the 6502's instructions under six different headings; Transfer, Arithmetic, etc. The instructions under each section are in a column on the left, with the modes of use and status flags affected across the top. The value printed in the chart is the hex for that particular opcode and the letters at the end under 'flags' indicate which flags are changed, and if they are set to a known state then the value is given in brackets. Table 2 is useful to convert these or other hex numbers into decimal. To convert a two digit hex number, locate the first hex digit in the first column and the second hex digit in the row across the top, the decimal equivalent is printed where the two meet in the table.

Having now covered all the 6502's instructions and getting a basic understanding of their functions, we can now move on, in the next issue, to some real programming examples which you will be able to type in, run, and modify.

Bye for now!

HEX TO DECIMAL CONVERSION

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

RULE: ROW FIRST THEN COLUMN
LOCATE FIRST HEX DIGIT IN ROW THEN SECOND IN COLUMN — READ OFF DECIMAL.

Table 2.

OPENING OUT

AN INTRODUCTION TO THE USE OF FILES Part 1

By Ron Levy

The primary task of any computer is the manipulation of data between devices, and it is how well organised and controlled this is that often differentiates a good, easy to use machine from a toy. I will try and explain how the Atari's I/O system functions, and give an overall insight into the design philosophy behind its layout.

Each of the many devices available to the programmer (and these include the screen editor, keyboard, and graphics display screen as well as the more obvious physical devices such as printers, cassette drives and disk drives) has its own set of machine code routines inside the operating system to carry out the instructions passed to them by your BASIC or machine code program. How then, does your program (or indeed the BASIC language) interface to these routines? It could have been done by having separate entry routines in the operating system for each individual command for each device. This would appear at first to be a simple system, but imagine writing a program to do a particular task which involved moving data to and from a device. If, later, you wanted to make the program perform the same task but on a different device you would need to re-write most of the program. Clearly, what is required is some form of common, overall link to access any device, and a universal set of commands and parameters. This is precisely the system which does exist in the Atari, and its heart is called the Central Input/Output routine, or CIO for short. Firstly though, lets take a look at files in general.

Suppose that we are writing a program which prints, at regular intervals, text data to a disk file. (It could equally, of course, be to cassette or printer!) The most obvious way of doing this would be by referencing the device and file name in each print command, for example: –

```
PRINT "TESTING",8,0,"D:file.dat"
```

This, as you can see, would be rather clumsy, especially when you consider that you would need to include all the 'mode' information as well. Not only would the program statements be awkward, but it would also mean that the device would have to search to locate the correct position in the file or device to dump the data on each command.

This is, however, the method which most novices to the field of programming seem to expect, and they often have trouble understanding how well organised machines such as the ATARI handle their files. So how should it be done?

Basically, the principle is that you declare, at the start of your program, which files you will be using, and associate a channel number (from 0 to 7) with it. This process is termed opening a channel for a file, and can be thought of as building an imaginary bridge between your program and the device/file you require. I will use this analogy to illustrate further points as we proceed. Inside your Atari there are eight of these imaginary bridges, or channels, but you can only use the seven numbered one to seven for your own files, since channel 0 is opened by the computer's operating system itself when it is switched on. It is opened to the screen display editor (E:), and this gives the 40*24 character screen, and allows the system to print to the screen if necessary.

For each I/O channel there are 16 bytes, or characters, reserved in memory for keeping information about a file that is being used. They are like note pads used to tell CIO what we want to be done to our file. These I/O Control Blocks (IOCB's), as they are called, would be used directly by you only if you were writing in machine language, since from BASIC this need not be of any concern as BASIC itself will deal with the IOCBs (and the jump to CIO) when you give the commands such as PRINT, INPUT, or OPEN. CIO can be thought of as a separate subroutine in the operating system which is used in machine code in exactly the same way as a GOSUB would be used in a well structured BASIC program.

BUILDING YOUR BRIDGE

The first step in using or creating your file is to create the imaginary bridge (channel) to the file, and from BASIC we do this with the OPEN command. There are, however, three important things which we need to tell the computer when we use this command. These are the following.

1. Filename

We need to give the exact name of the file and the name of the device that the file is to be found on. We specify the device first by giving the device's assigned name. Each has its own unique one character reference, E for the screen editor (for graphics 0 text), K for the keyboard, S for the graphics display (for graphics 1 and above), C for the cassette, P for the printer, and D for the disk drives. To signify the device name we also place a colon afterwards, for example, we would say C: for cassette, or D: for disk. A further complication, though, is that we could have several of the same device available – perhaps we have two disk drives (sheer luxury!) connected to the computer. We would need to specify which drive we want to refer to for the file and we can do this by simply saying D1: or D2: (you can have up to four disk drives connected to your computer if you are wealthy enough!) If we do not quote a number for the drive then the operating system assumes that you mean device number 1. This principle of an assumption being made by the computer is called a default, i.e. we say the device number defaults to 1. The file name

It is a sad fact that one of the most impressive features of the Atari computer is perhaps the least fully understood, and is certainly the least well explained. It is the Input/Output.

itself is only required with disk drives since they are the only devices which allow multiple named files. If you do quote a file name where one is not required, then CIO will simply ignore it rather than produce an error message, for example, "E:DATAFILE" will be treated as "E:", and "C:DATAFILE" would be considered as just "C:".

2. The Channel Number

As explained earlier, we need to chose one of the seven available channels, or bridges, for our data to be passed through. We indicate the term channel in our program by using the hash symbol #, i.e. we would refer to channel 1 as 'hash one', and would write it as #1.

3. Mode and Data Direction

We will need to tell CIO how we want the channel opened and the most important aspect is the data direction, i.e. whether we want to send data to the device, get data from the device, or even both. We do this with the first of the two bytes provided. Basically, a 4 here means "I only want to read data from the device/file", and an 8 means "I only want to send data to the device/file". The terms we use are "open for input", and "open for output" respectively.

Here is a typical open command:–

OPEN #1,8,0,"D:FILENAME"

We have the command OPEN, followed by the channel number #1. Then there are two bytes which allow auxilliary information to be passed through to the operating system, and you can see that I have only used the first one, the data direction byte, and it is set to "open for output". As a general rule, the first byte must always be used, and its value is usually independent of the device we are refering to since 8 always means "open for output" and 4 always means "open for input", regardless of the type of device. For example, if we decide to open the printer, then we will naturally be opening it for output, since you cannot get data from a printer, and so we would use the value 8.

The second byte is there for when you need to give more special instructions to the device to tell it precisely how to handle the data to be transfered, for example, if you were using the old thermal printer there is a value which can be placed into this second byte which will make it print sideways.

Last is the filename, and the point to notice here is that, because it is just a collection of ASCII characters, it is treated and held by BASIC just as it would a normal string. The advantage here is that you could actually use a string variable, one which was perhaps decided upon in a different part of the program, or perhaps even INPUT from the user at the keyboard as a string. For example:–

```
10 DIM A$(20)
20 INPUT A$
30 OPEN #1,8,0,A$
40 PRINT #1;"TESTING"
50 CLOSE #1
```

The user may type C: for cassette, D1:filename.dat for disk drive 1, P: for printer, or E: for the screen, and so on.

CROSSING YOUR BRIDGE

Now that we have seen how to build 'bridges' to files/devices, let us see how we can transfer data across them. There are two methods used by the Operating System for transfering data through its CIO routine to and from files. The first is the simplest and this is called Byte Transfer, where data is sent and received without any significance being attached to any byte values, i.e. carriage return (155) has no special function. This makes it ideal when the data bytes may have any random value from between 0 and 255. The second method is called Record Transfer, and here we deal with the data in terms of text strings, i.e. a stream of printable characters with an End Of Line code (155) on the end, the same code as that generated when you press the RETURN key on your keyboard. Let us now see how and when we should use these two methods from BASIC.

1. Byte Access

To use this method from BASIC we use the commands GET and PUT. Suppose that we have a file open on channel 1 and we want to output a single byte or character through it of value 65. We would place the following command into our program:–

PUT #1,65

We could of course replace the number 65 with a number variable, such as X, but it must be remembered that PUT will only send a single byte, and so the number is restricted to a value of between 0 and 255. To retreive a single byte back from a file we use the complementary command, GET. This would take the form of:–

GET #1,X

2. Record Access

Here, rather than single bytes or characters, we can transfer data in the form of strings of characters. The command we use to send data in BASIC is the PRINT command, and this works in exactly the same way as we would use it to print a string to the screen! The only addition we must make is to include some reference to the channel we want to send to, for example:–

PRINT #1;A$

This will print A$ to channel 1. Indeed it could be said that BASIC treats the PRINT #1 in the same way as PRINT because it assumes that when we print to the screen we want to print to channel zero– remember that earlier we said that the Operating System OPEN's the screen editor on this channel when you first switch your computer on. Another example, perhaps, of a default, i.e. missing the channel number on the PRINT command means take the default of PRINT #0. The same principles also apply to the INPUT command. If our program gives the command:–

INPUT #1,A$

Then the operating system will read from the file open on channel one, a character at a time and start placing them into A$. It will only stop when either the end of the file is reached (when error 136 is returned) or when the END OF LINE character is reached – this is character 155, the RETURN key code. This is why when you give the INPUT A$ or INPUT #0,A$ command to get a string from the screen the computer will not stop accepting characters until you press the RETURN key.

Let us try a few examples. Type in and RUN Program 1, but on line 100, where you will find the OPEN command, I have used the cassette drive as the output file. If you own a disk drive then you may replace C: with D:TEST, and you must have switched your computer on with a DOS master disk in your drive.

```
1 REM ********************************
2 REM *      Demo Program 1.         *
3 REM * Creates a small text file *
4 REM * onto a cassette.  Change  *
5 REM * the OPEN command for use  *
6 REM * with a disk drive.        *
7 REM ********************************
10 DIM A$(100)
100 OPEN #1,8,0,"C:"
110 PRINT #1;"This data file"
120 PRINT #1;"Consists of text."
130 PRINT #1;"It was created"
140 PRINT #1;"using the PRINT"
150 PRINT #1;"Command."
200 CLOSE #1
```

If you used a cassette drive for Program 1 then a beep will prompt you to press PLAY and RECORD, just as when you CLOAD a program. You should rewind the tape when the READY prompt appears. You have now created a small text file, and the next step is to examine this file using each of the data transfer methods explained earlier. Type NEW, and enter Program 2.

```
1 REM ********************************
2 REM *      Demo Program 2.         *
3 REM * Reads back the file which *
4 REM * we created with program 1.*
5 REM * It uses the INPUT command.*
6 REM ********************************
10 DIM A$(100),B$(1)
100 OPEN #1,4,0,"C:"
110 INPUT #1,A$
120 PRINT A$,"<RETURN>";
130 INPUT B$
140 GOTO 110
```

Every time you press RETURN the program inputs another string from the file and prints it to the screen. Now try Program 3, remembering once again to rewind your tape if you are using cassette. Notice that this time the program reads the file one character at a time. You should also see that there are several bytes which do not print a character but cause the cursor to jump down to the next line on the

screen. These are the RETURN key codes we mentioned earlier, and the important aspect to note is that in this program, using the GET command, they have no particular significance. They are simply read in just as any other character would be.

```
1 REM ********************************
2 REM *      Demo Program 3.         *
3 REM * Reads back the file which    *
4 REM * we created with program 1.   *
5 REM *  It uses the GET command.    *
6 REM ********************************
10 DIM B$(1)
100 OPEN #1,4,0,"C:"
110 GET #1,A
120 PRINT "    Byte Value ";A,
130 PRINT "Character ";CHR$(A);
140 PRINT " ";:INPUT B$
150 GOTO 110
```

Well, now we have seen how to create a small file and read it back at a later date. A very simple operation, of course. But lets now see in a little more detail how the computer transfers our data. When you RUN Program 1 to create your file, if you were to pause the computer between each PRINT instruction you would notice that the computer does not actually appear to send each string as it is printed to the file. What is happening is that the computer is using a BUFFER in its memory to catch each string you print, and when this buffer gets full, only then will it be sent to the cassette or disk drive. This might at first seem a little strange, but think for a moment about the benefits. If a buffer was not used and, (as happens with the screen display) each string was sent to the file/device as it was printed your cassette would be continually starting and stopping– a waste of time as well as causing unnecessary wear.

The size of the buffer varies between different devices, but there are 128 bytes, or characters if you like, in the cassette buffer and 125 in the disk buffers. Every beep you hear when you are saving or loading a program, (or reading or writing to a file) is one of these buffer loads of characters being transfered between your computer and the external storage device.

It is at this stage that I really should mention the problem caused by a magnificent goof by the people who wrote the Atari Operating System! The bug they left us is that when you OPEN the cassette drive for output (to create a new file) the system will immediately start the cassette motor, and will send 15 seconds of tone, called the leader. This is quite normal and proper – you need not concern yourself with why – but what happens next is certainly not! After this the computer continues with your program as expected, but it forgets to stop the cassette motor and switch off the tone! Subsequent buffers filled by your program will be sent quite normally, but if your program does not happen to send the first one for some time then you may have a very large gap, making the file impossible to read back normally later. Fear not for there are in fact two solutions. The first is to make sure that at least 128 characters of data are output to the cassette file as soon as it is OPENed, i.e. if you are writing a program which, for example, dumps a large block of memory to cassette you should place the OPEN command just before the output routine rather than at the beginning of the program.

This is however not always possible since some of your programs may only want to send data to the cassette in short, occasional lumps. Here, we must use a different method. Since the bug only occurs before your first 128 byte data block, the answer is to print a dummy string to the file (immediatley after opening it) with a length of at least 128 bytes. This ensures that the cassette drive from then on sits in the idle state until it is required to receive further buffers, but it does mean that you must make sure that the program which later reads back from this file performs an input and discards this string. Programs 4 & 5 show how to implement this dummy print method.

```
0 REM ********************************
1 REM *      Demo Program 4.         *
2 REM * Creates a cassette file      *
3 REM * and fills the first buffer*
4 REM * with dummy data to force     *
5 REM * the cassette drive to stop*
6 REM * so that the program can      *
7 REM * print to it at as slow a     *
8 REM * pace as required.            *
9 REM ********************************
10 DIM A$(100)
100 OPEN #1,8,0,"C:"
110 FOR I=1 TO 127
120 PRINT #1;"X";
130 NEXT I
140 PRINT #1
150 REM That's the first bufer sent!
160 REM
200 PRINT "Type whatever you wish"
210 PRINT "  END will terminate."
300 INPUT A$
310 IF A$="END" THEN 900
320 PRINT #1;A$
330 GOTO 300
400 REM
900 CLOSE #1
```

```
0 REM ********************************
1 REM *      Demo Program 5.         *
2 REM * Reads back the file which    *
3 REM * we created with program 4.   *
4 REM *    Notice that its first     *
5 REM * task is to read the dummy    *
6 REM * string and discard it.       *
7 REM ********************************
8 REM
9 REM
10 DIM A$(100)
100 OPEN #1,4,0,"C:"
110 INPUT #1,A$
200 INPUT #1,A$
210 PRINT A$
220 GOTO 200
```

CLOSING STAGES

When you have finished sending data to a file there is one function still left to perform and this is the CLOSE. With this command we tell the system that we no longer wish to send or receive data to or from our file, and the two important tasks that the computer will then perform are as follows: –

1. Send the Last Buffer

Previous PRINTS or PUTS to the file may not have filled the buffer, so the system sends what is there and it is stored on the cassette as a partly filled record. This obviously only applies to files that have been OPENed for output.

2. Release the Channel

After the CLOSE we can re-use the same channel (or bridge as we have been calling it) for another file. This is important since we only have five channels (numbered 1-5) available to use in our programs. Although the system does in fact have eight, it uses channel 0 for text display (graphics 0), channel 6 for graphics 1 and above displays (when you call them with the graphics command), and channel 7 is used by BASIC in some of its commands, such as LPRINT, CSAVE, LOAD, RUN, CLOAD, etc. (Yes – even the BASIC language itself must obey the same rules as you where data transfers and files are concerned!!)

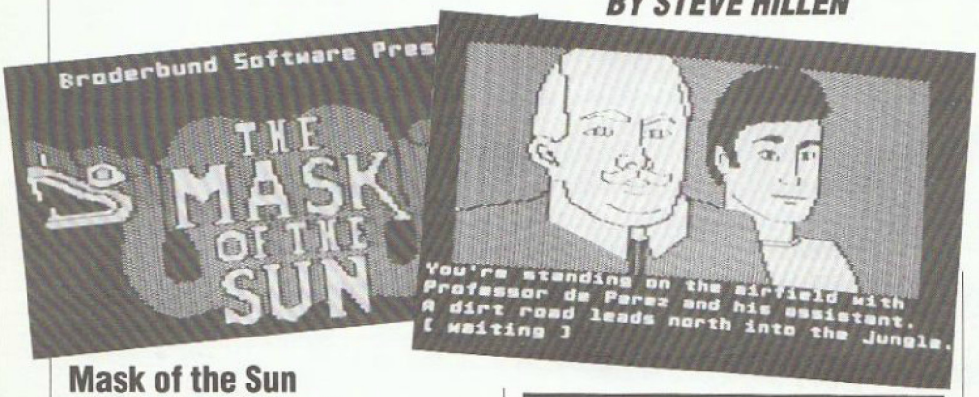
IN CONCLUSION

A final word now for those of you who do not yet understand the purpose of a buffer. Let us imagine that you are standing on one side of a long bridge, and at random intervals someone gives you a couple of bricks which need to be taken to the other side. You could of course dash back and forth across the bridge each time you are given the bricks, (be it 1, 2 or 15 bricks) but this would soon wear you out, and would take a considerable amount of time. Would it not be far better to borrow a wheelbarrow and wait until enough people have given you bricks for you to fill the barrow, and then take a trip across with a full load? This is a very good comparison to the use of buffers – the operating system keeps a section of memory for each disk or cassette file open, and when this becomes full it sends it out to the file. The same also occurs, but in reverse, when you are reading from a file. Since your program may only want tiny sections at a time, the computer brings in a large chunk to fill its buffer, and only re-fills it from the file when it becomes empty.

Hopefully you should now have a clearer picture of how files are accessed on your Atari, and perhaps be able to experiment a little. In the next issue we will show what goes on in a little more detail, and illustrate some of the special functions of your cassette and disk drives.

ADVENTURE INTO THE ATARI

BY STEVE HILLEN



Mask of the Sun

48K disk from Broderbund

This adventure occupies 2 double sided disks, so is quite large as graphics adventures go. You play the part of Mac Steel, a sort of Indiana Jones type of archaeologist who has unfortunately been affected by a mysterious gas released on tinkering with a South American trinket. If a cure is not found then Mac will die – heavy exertions must be avoided !

The game starts as you arrive in Aztec country, and as you leave the plane, the graphics come alive. Usually, two or three full screen colour pictures are flipped onto the screen, giving a good impression of moving into the scene. In this case, Raoul and the Professor become larger as you approach them. The movement is naturally coarse, but it's a good effect all the same.

Soon you are travelling in the rain forest, visiting any of the 3 ruined temples in search of the Mask of the Sun, an artifact which, according to legend, holds the cure to your illness. Animation is used to excellent effect, such as the witch, human sacrifice, skeletons, and especially the snake. Other hazards include large unbalanced boulders, hidden pits and a robber. As you wander around the inside of the temples, another feature of the graphics is shown – the view of corridors change as you approach from different directions, just as in real life. Sound effects are also used to good effect – birds chirp, snakes hiss, and doors crash shut. One very original feature is that in certain situations, you will need to type in responses quickly to avoid death, so be warned !

Broderbund have come up with a terrific adventure here, ranking with the hardest, but also with the liveliest and most imaginative. Buy it !

I'm glad to say that I've had a rather larger number of letters this time — thanks to all those who sent in answers, questions & suggestions for the column. Once again, the column follows the same format of reviews and clues, although I'm open to other suggestions.

Sorceror

32K disk from Infocom
Reviewed by Gary Cheung

As I boldly leaped into the world of Sorceror, I found myself in a dark, twisted forest and facing me was a blood-thirsty Hellhound. My pulse quickens and without waiting for it to tear me apart, I ran for the nearest tree. My heart nearly jumped into my mouth as I almost fell out of the tree (dining room chair actually) as a boa-constrictor came slithering towards me. Running like a possessed soul, I discovered that a cloud of blood-sucking insects were just looming over the horizon! As I panicked and ran for cover, I stepped on a magic mine and kaboomm!!! I woke up with a cold sweat.

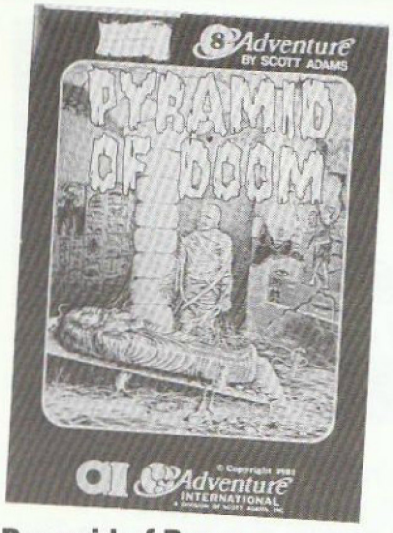
Having recovered my wits, I soon discovered that Belboz the Necromancer, grand and powerful leader of the Guild of Enchanters has disappeared under suspicious circumstances. After some investigations, it became obvious that he is in great danger, and so is the Kingdom and everything as I know it! With my trusty spell-book, and a strange vial that arrives a day before you order it, I embarked on a perilous and fantastic journey into the land of my nightmares!!

After visiting the river monsters and walking round the fortress, I soon found myself stumbling around in a three dimensional glass maze. Having struggled to the other side, I found that not only could I not take my due reward with me but I had to re-enter the maze in a hurry (the reason will be painstakingly obvious once you are there). Then I found the maze had changed since I came through it !!

The visit to the amusement park was both fun and rewarding.Having livened things up a bit in the carving room, I gave myself a helping hand and went down to a lagoon shore.I noticed that my Bellboz-Seeking-Device (Infocom call it something else) was glowing brightly ....

Sorcerer is the follow up to the brilliant Enchanter. In both adventures, you have no treasure to find but a quest to fulfil, and believe me, that's enough. In order to complete your quest, you will need to learn and cast spells. In Sorcerer you have a few spells to start you off and as in Enchanter, some more can be found on the way, but you will have to work for them! You will also find some magic potions, of which I particularly enjoyed the "Exquisite Torture". Just for fun, I animated a floor polisher, read the mind of the sleepy gnome and brought the Emporer's statue to life. Casting the self-resurrection spell inside the glass maze also has an interesting effect.

Sorcerer is rated at ADVANCED level but I found it slightly less demanding than Enchanter. It is an excellent adventure and up to Infocom's usual high standards. If you have played and enjoyed Enchanter, Sorcerer is a MUST. If you have not yet played Enchanter then get it, and while you are at it, get Sorcerer as well !



Pyramid of Doom

24K cassette by Scott Adams

This is the eighth adventure in Scott's original series and is one of my favourites, the whole game being a very devious treasure hunt. As an intrepid explorer you start in the desert near a pyramid, the first problem being to get inside without being pulped by a large stone block above the doorway. Once inside however, there are thirteen treasures to collect, and also a place must be found to keep them. Hungry rats, a sleepless mummy, and an evil pharaoh are all out for your blood, but the wandering nomad who follows you about can be just as dangerous. There is a lot of action and there are many problems to be solved inside the pyramid, and Scott has put in several rather nasty red herrings, especially the camel jerky !

Overall, I liked this adventure because it has so many varied characters and rooms in it, and by solving one problem, a whole new region may be opened up for exploration. It gives great satisfaction to triumph in the end – but you'll need to know a little about ancient Egypt (or horror movies) to succeed – it's well worth a go.

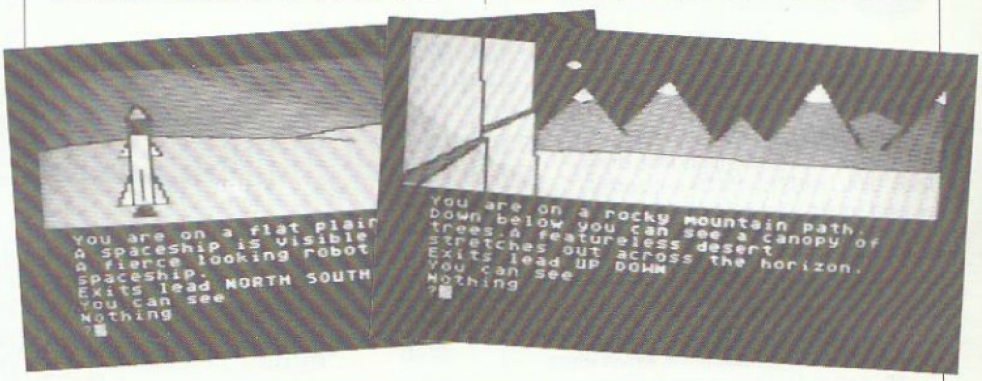
Stranded

32K cassette from English Software

This adventure has the distinction of being the only cassette based graphics adventure for the Atari. As Special Agent Sid (good grief) you need to escape from the mysterious planet back to Earth (shades of Strange Odyssey?). The graphics are outlined then filled in while you watch. It beats me why this has not been done before for the Atari, as it's very common on the Spectrum. The graphics themselves are simplistic but effective, and overall a lot of hard work has gone into this program.

There are many locations with their pictures being complemented by a few lines of text. As per usual, there are quite a few little problems to be solved before the spaceship is ready to launch, and in this respect, the game is not found lacking.

Overall then, this is quite a good little adventure, the weak storyline and silly names detracting somewhat. If you are used to the complexities of Zork or Level 9 adventures, then you may be disappointed.



Letters Section

Hugh Denholm has suggested that we might start a service for people to sell their completed adventures. I'd be interested in any other comments.

Thanks in particular to Peter Lister for all his answers and special thanks go to Gary Cheung for supplying an excellent map of Snowball, and for the many cryptic clues he sends in !

Don't forget that this is your column, and if you have any queries about how to play or even write adventures, or just suggestions for the column then write in. Reviews of any of your favourite adventures would also be welcome, as would solutions to any of the questions below.

As in Issue 7, there is a small decoder to decipher the cryptic clues below. Just type in lines 10,20 and 40 below, then add the line 30 of your choice, type RUN and get the clue. This will hopefully prevent you from seeing unwanted clues.

```
10 DIM TEXT$(100),MC$(21):FOR A=1
TO 21:READ D:MC$(A)=CHR$(D):
NEXT A
20 DATA 104,104,104,168,136,104,133,
204,104,133,203,177,203,73,6,145,203,
136,16,247,96
40 X=USR(ADR(MC$),LEN(TEXT$),
ADR(TEXT$)):?CHR$(125):?:?TEXT$
```

Answered Questions

Savage Island Pt 2

What do you do after the password ?

30 TEXT$="DTCGRN&BCCVJ_(&AI
&@OCJB(&DTCGRN&ISR("
by Peter Lister

Planetfall

How do you repair the communicator?
30 TEXT$="_CJJIQ&JOANR&RCJJU
&_IS&DSRRIH&@IT&EITTCER&
ENCKOEGJ"
by Gary Cheung

Starcross

The blue rod.
30 TEXT$="7&BOUM&SHBCT&
UVNCTC+RSTH&RI&4+VSR&USOR
&IH&RIV+RSTH&RI&5&RNCH&2+
TIB&IH&IRNCT&BOUM"
by Gary Cheung and Hugh Denholm

Where is the silver rod ?
30 TEXT$="C^GKOHC&RNC&ASH&
PCT_&EGTC@SJJ_"

Sands of Egypt

How does the camel move ?
30 TEXT$="@CCB&OR&QORN&
BGRCU&RNCH&KISHR"

How do you get the palm fronds ?
30 TEXT$="SUC&RNC&G^C&@TIK&
RNC&V_TGKOB"

How do you get out of the
underground chamber ?
30 TEXT$="SUC&RNC&JGBBCT&
GHB&ROKC&OR&QCJJ"
by Mr A. Lusher

Golden Voyage

Trouble making second stone tablet ?
30 TEXT$="4&OH&TSDDJC&O@&
URGRSC+TIIK&I@@&UCETCR&
VGUUGAC+BOA&BIQH&RQOEC&
HCGT&EGPC"
by Mr P. Withers and Gary Cheung

Adventureland

How do you get past the bear without
giving the honey ?
30 TEXT$="UNISR'"

Zork II

How do you get past the dragon?
30 TEXT$="NOR&OR&QORN&
UQITB&GHB&JCGB&OR&RI&OEC"
by Gary Cheung

How do you get the key off the
unicorn?
30 TEXT$="VTOHECUU'"
by Gary Cheung

Arrow of Death Pt 2

Can't cross water?
30 TEXT$="QCGT&QGTTOIT&
SHO@ITK+RSTH&OTIH&QNCCJ+
LSKV&QORN&MORC&@TIK&QOHB
_&TIIK"
by Peter Lister

Unanswered Questions

Savage Island Pt 2

Can't kill dinosaurs nor change back from neanderthal.

Escape from Pulsar 7

Can't remove cups from captain's cabin ceiling.

Feasability Experiment

Can't get past guard in mine.

10 Little Indians

Can't do anything with couch or safe.

QUICKPLOT

A FAST GRAPHICS 8 PLOT-DRAWTO HANDLER

BY PETER BLACKMORE

So far, there have been quite a number of Graphics 8 utilities presented in the magazine. These include an 80 column screen, text on Graphics 8, a data compressor, and a routine to print out the screen. With all these improvements, it's about time the Plot-Drawto function was brought up to par!

The Atari operating system does not use a particularly fast method to draw lines, a far better one is Bresenham's algorithm, devised in 1965. The way this algorithm works is best illustrated by example.

Suppose we have just Plotted (0,0) and we wish to Drawto (300,150). The Drawto function must calculate all the points between the two limits which must be filled in. It does this in the following manner.

The change in X and the change in Y, deltax and deltay respectively, are calculated. In this case they are 300 and 150. Since deltax > deltay, each time the X value of the point currently being plotted is altered, the Y co-ordinate may or may not change. In this case Y changes once for every two times X changes, see Diagram 1.

Bresenham's method keeps an error term to which is added the slope of the line being drawn (deltay/deltax) on each iteration. The sign of this error term then indicates whether the drawn line is above or below the ideal line, and hence whether or not the Y value should be changed.

The program listed below is an implementation of this algorithm, and used from Basic will draw lines in half the time that the normal Plot-Drawto takes. Listing 1 is for use from Basic and includes a number of demos. The Plot-Drawto is accessed by: –

A=USR(P,X,Y) for PLOT X,Y
A=USR(D,X,Y) for DRAWTO X,Y

P and D are set to 1536 and ADR(DRAW$) respectively. Remember to save a version out before running the program, and also that this routine will not detect errors such as PLOT 700,400, so make sure that you pass it the correct values or suffer the consequences!

Listing 2 is the assembly code for the Plot-Drawto function for all of you that want to speed it up even more, there are ways of doing this! If you know of a different method why not send it in!!

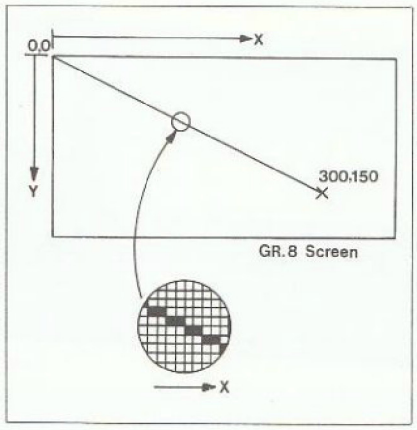


Diagram 1.

Listing 1.

```
1 DIM DRAW$(392)
2 FOR A=1 TO 392:READ OP:DRAW$(A,A)=CH
R$(OP):NEXT A
3 FOR A=1536 TO 1694:READ OP:POKE A,OP
:NEXT A
5 P=1539:D=ADR(DRAW$):DEL=300
10 GRAPHICS 24:POKE 709,11:POKE 710,0:
COLOR 1
20 FOR A=0 TO 319 STEP 2:X=USR(P,159,0
):X=USR(D,A,191):NEXT A
30 GOSUB DEL
40 FOR A=0 TO 319 STEP 3:X=USR(P,159,1
91):X=USR(D,A,0):NEXT A
50 GOSUB DEL
60 FOR A=0 TO 191 STEP 2:X=USR(P,0,95)
:X=USR(D,319,A):NEXT A
70 GOSUB DEL
80 FOR A=0 TO 191 STEP 3:X=USR(P,319,9
5):X=USR(D,0,A):NEXT A
90 GOSUB DEL
100 PLOT 8,8:DRAWTO 311,8:DRAWTO 311,1
83:DRAWTO 8,183:DRAWTO 8,8
110 FOR A=95 TO 10 STEP -2:X=USR(P,159
,95):X=USR(D,10,A):NEXT A
120 FOR A=10 TO 309 STEP 3:X=USR(P,159
,95):X=USR(D,A,10):NEXT A
130 FOR A=10 TO 181 STEP 2:X=USR(P,159
,95):X=USR(D,309,A):NEXT A
140 FOR A=309 TO 10 STEP -3:X=USR(P,15
9,95):X=USR(D,A,181):NEXT A
150 FOR A=181 TO 96 STEP -2:X=USR(P,15
9,95):X=USR(D,10,A):NEXT A
160 GOSUB DEL
170 FOR A=90 TO 158:X=USR(P,A,A):X=USR
(D,319-A,A):X=USR(D,319-A,191-A):X=USR
(D,A,191-A):X=USR(D,A,A):NEXT A
180 GOSUB DEL
190 FOR A=0 TO 95:X=USR(P,A,A):X=USR(D
,319-A,A):X=USR(D,319-A,191-A):X=USR(D
,A,191-A):X=USR(D,A,A):NEXT A
200 GOSUB DEL
210 FOR A=94 TO 0 STEP -3:X=USR(P,A,A)
:X=USR(D,319-A,A):X=USR(D,319-A,191-A)
:X=USR(D,A,191-A):X=USR(D,A,A):NEXT A
220 GOSUB DEL
299 END
300 FOR A=1 TO 500:NEXT A:? #6;CHR$(12
5);:RETURN
999 END
1000 DATA 104,201,2,208,118,104,141,16
8,6,104,141,167,6,104,208,107,104,141,
169,6,216,169,64,141,165,6
1010 DATA 141,166,6,169,0,141,163,6,14
1,164,6,173,167,6,56,229,85,141,170,6,
173,168,6,229,86,16
1020 DATA 24,14,165,6,73,255,141,171,6
,173,170,6,73,255,24,105,1,141,170,6,1
73,171,6,105,0,141
1030 DATA 171,6,173,169,6,197,84,144,7
,56,229,84,160,0,240,9,14,166,6,165,84
,56,237,169,6,141
1040 DATA 172,6,173,171,6,208,8,173,17
0,6,205,172,6,144,7,169,0,240,51,76,77
,160,173,172,6,74
1050 DATA 133,212,56,173,170,6,229,212
,141,173,6,173,171,6,233,0,141,174,6,1
73,174,6,48,51,165,85
1060 DATA 172,165,6,48,15,24,105,1,133
,85,165,86,105,0,160,0,240,11,240,75,5
6,233,1,133,85,165
1070 DATA 86,233,0,133,86,173,173,6,56
,237,172,6,141,173,6,173,174,6,233,0,1
41,174,6,173,173,6
1080 DATA 24,109,170,6,141,173,6,173,1
74,6,109,171,6,141,174,6,164,84,200,17
3,166,6,16,2,136,136
```

```
1090 DATA 132,84,32,21,6,238,163,6,173
,163,6,205,172,6,208,155,96,173,171,6,
74,133,213,173,170,6
1100 DATA 106,133,212,173,172,6,56,229
,212,141,173,6,169,0,229,213,141,174,6
,173,174,6,48,31,164,84
1110 DATA 200,173,166,6,16,2,136,136,1
32,84,173,173,6,56,237,170,6,141,173,6
,173,174,6,237,171,6
1120 DATA 141,174,6,173,173,6,24,109,1
72,6,141,173,6,173,174,6,105,0,141,174
,6,165,85,172,165,6
1130 DATA 48,13,24,105,1,133,85,165,86
,105,0,160,0,240,9,56,233,1,133,85,165
,86,233,0,133,86
1140 DATA 32,21,6,238,163,6,208,3,238,
164,6,173,163,6,205,170,6,208,152,173,
164,6,205,171,6,208,144,96
1150 DATA 76,77,160,104,201,2,208,248,
104,133,86,104,133,85,104,208,239,104,
133,84,216,165,85,141,161,6
1160 DATA 165,86,141,162,6,169,0,133,2
13,168,78,162,6,110,161,6,42,78,161,6,
42,78,161,6,42,170
1170 DATA 165,84,133,212,6,212,38,213,
6,212,38,213,6,212,38,213,165,212,141,
159,6,165,213,141,160,6
1180 DATA 6,212,38,213,6,212,38,213,16
5,212,24,101,88,133,212,165,213,101,89
,133,213,165,212,24,109,159
1190 DATA 6,133,212,165,213,109,160,6,
133,213,165,212,24,109,161,6,133,212,1
65,213,109,162,6,133,213,165
1200 DATA 200,240,8,177,212,29,151,6,1
45,212,96,189,151,6,73,255,49,212,145,
212,96,128,8,32,2,64,4,16,1
```

Listing 2.

```
00010 ; FAST GRAPHICS 8 PLOT & DRAWTO
00020 ;
00030 ; Written on the SynAssembler.
00040 ;
00050 ;
00060 ; OS Equates
00070 ;
00080 SAVMSC   .EQ $58
00090 ROWCRS   .EQ $54
00100 COLCRS   .EQ $55
00110 X        .EQ COLCRS
00120 Y        .EQ ROWCRS
00130 ;
00140 ; Temporary workspace
00150 TEMP     .EQ $D4
00160 ;
00170 COLOUR   .EQ $C8 ;set by COLOR 0
00180 ;                       or 1
00190 ;
00200 ; format for draw command
00210 ;
00220 ; A=USR (START,DRAWX,DRAWY)
00230 ;
00240 ; on entry x,y = plotx,ploty
00250 ;
00260 START    PLA     ;no. of
00270          CMP #2  ;   parameters
00280          BNE ERR ;only 2 wanted
00290 ;
00300          PLA     ;Hi of draw x
00310          STA DRAW.X+1     point
00320          PLA     ;Lo of draw x
00330          STA DRAW.X       point
00340          PLA
00350          BNE ERR ;Hi byte must
00360          PLA     ;Lo byte  be 0
00370          STA DRAW.Y
00380          CLD     ;BASIC set this
00390 ;
00400 ;
00410 ; Initialise variables
00420 ;
00430 ; Note use of $40 to represent
00440 ;  a positive x or y increment
00450 ;  When this is shifted left
00460 ;  once the negative flag is set
00470 ;
00480 ;
00490 CLEAR.VAR
00500          LDA #$40  ;positive
00510          STA NEG.X ;to begin
00520          STA NEG.Y ;with
00530 ;
00540          LDA #0    ;zero loop
00550          STA I.LOOP     counter
00560          STA I.LOOP+1
00570 ;
00580 ;
00590 ; Find out the x increment
00600 ;  -> delta.x
00610 ; and its sign
00620 ;  -> neg.x
00630 ;
00640 ; DELTA.X = ABS(DRAW.X-PLOT.X)
00650 ; NEG.X   = SGN(DRAW.X-PLOT.X)
00660 ;
00670 ;
00680          LDA DRAW.X  ;Lo byte
00690          SEC     ;subtract last
00700          SBC X   ;point plotted
00710          STA DELTA.X ;save
00720          LDA DRAW.X+1
00730          SBC X+1 ;test sign of
00740 ;                MSB and set flag
00750          BPL .1
00760 ;
00770          ASL NEG.X   ;make neg
00780          EOR #$FF    ;and take
00790          STA DELTA.X+1
00800          LDA DELTA.X ;two's
00810          EOR #$FF    ;complement
00820          CLC         ;to obtain
00830          ADC #1      ;positive
00840          STA DELTA.X ;magnitude
00850          LDA DELTA.X+1
00860          ADC #0      ;maintain
00870 .1       STA DELTA.X+1
00880 ;                    Hi byte also
00890 ;
00900 ;
00910 ; Repeat the above to obtain
00920 ;  sign and magnitude of y
00930 ;  increment
00940 ; Note this is simpler than for x
00950 ;  since the maximum value of y
00960 ;  must always be less than 255
00970 ;  and hence fits into one byte
00980 ;
00990 ;
01000 ; DELTA.Y = ABS (DRAW.Y-PLOT.Y)
01010 ; NEG.Y   = SGN (DRAW.Y-PLOT.Y)
01020 ;
01030 ;
01040          LDA DRAW.Y
01050          CMP Y
01060          BLT .2
01070 ;
01080          SEC
01090          SBC Y
01100 ;
01110          LDY #0   ;unconditional
01120          BEQ .3   ;  branch
01130 ;
01140 .2       ASL NEG.Y ;set negative
01150          LDA Y    ;and calculate
01160          SEC         ;magnitude
01170          SBC DRAW.Y
01180 .3       STA DELTA.Y ;save
01190 ;
01200 ; is delta.y > delta.x
01210 ;
01220          LDA DELTA.X+1
01230          BNE .4
01240          LDA DELTA.X
01250          CMP DELTA.Y
01260          BLT X.LT.Y
01270 ;
01280 .4       LDA #0     ;a pseudo
01290          BEQ TRANSFER ;jump
01300 ;                    ;instruction
01310 ERR      JMP $A04D  ;on error
01320 ; to BASIC's warmstart address
01330 ;
01340 ;
01350 ; come here if
01360 ;  delta.y > delta.x
01370 ;
01380 ; Initialise value of error term
01390 ;
01400 ; ERROR := DELTA.X - DELTA.Y/2
01410 ;
01420 X.LT.Y
01430          LDA DELTA.Y
01440          LSR      ;divide by 2
01450          STA TEMP ;save for later
01460          SEC
```

```
01470           LDA DELTA.X
01480           SBC TEMP    ;get error
01490           STA ERROR   ;save
01500           LDA DELTA.X+1
01510           SBC #0      ;and Hi byte
01520           STA ERROR+1
01530 ;
01540 ;
01550 ; FOR I.LOOP := 1 TO DELTA.Y DO
01560 ;
01570 REPEAT.LOOP
01580 ;
01590 ;
01600 ; is ERROR > 0
01610 ;
01620           LDA ERROR+1
01630           BMI E.LT.0
01640 ;
01650 E.GT.0
01660 ;
01670 ; X := X NEG.X 1
01680 ;
01690 ; ie if neg.x is negative then
01700 ;  x=x-1
01710 ;  otherwise
01720 ;  x=x+1
01730 ;
01740           LDA X
01750           LDY NEG.X
01760           BMI SUB1    ;x negative
01770 ;
01780           CLC         ;add one
01790           ADC #1      ;to current
01800           STA X       ;point to
01810           LDA X+1     ;plot
01820           ADC #0
01830 ;
01840           LDY #0   ;unconditional
01850           BEQ STORE1 ;   jump
01860 ;
01870 ;
01880 TRANSFER BEQ X.GT.Y  ;used to
01890 ;         jump down the program
01900 ;
01910 ;
01920 SUB1      SEC
01930           SBC #1      ;subtract
01940           STA X       ;one
01950           LDA X+1
01960           SBC #0
01970 STORE1    STA X+1
01980 ;
01990 ;
02000 ;Adjust error term as follows
02010 ;
02020 ;ERROR := ERROR+DELTA.X-DELTA.Y
02030 ;
02040 ;
02050           LDA ERROR
02060           SEC
02070           SBC DELTA.Y
02080           STA ERROR
02090           LDA ERROR+1
02100           SBC #0
02110           STA ERROR+1
02120 ;
02130 ;
02140 ; ERROR less than zero
02150 ;
02160 E.LT.0
02170 ;
02180 ; ERROR := ERROR +DELTA.X
02190 ;
02200 ;
02210           LDA ERROR
02220           CLC
02230           ADC DELTA.X
02240           STA ERROR
02250           LDA ERROR+1
02260           ADC DELTA.X+1
02270           STA ERROR+1
02280 ;
02290 ;
02300 DO.Y
02310 ;
02320 ; similar to x
02330 ; Y := Y NEG.Y 1
02340 ;
02350           LDY Y
02360           INY
02370           LDA NEG.Y
02380           BPL OK1
02390 ;
02400           DEY
02410           DEY
02420 ;
02430 OK1       STY Y
02440 ;
02450 ;Now plot the point in x,y
02460 ;
02470           JSR PLOT.POINT
02480 ;
02490           INC I.LOOP  ;increment
02500           LDA I.LOOP  ;counter
02510           CMP DELTA.Y
02520           BNE REPEAT.LOOP
02530 ;
02540           RTS    ;back to BASIC
02550 ;
02560 ;
02570 ; come here if
02580 ; delta.x > delta.y
02590 ;
02600 ;
02610 X.GT.Y
02620 ;
02630 ;initialise error term
02640 ;
02650 ; ERROR := DELTA.Y - DELTA.X/2
02660 ;
02670           LDA DELTA.X+1
02680           LSR
02690           STA TEMP+1
02700           LDA DELTA.X
02710           ROR
02720           STA TEMP
02730 ;
02740           LDA DELTA.Y
02750           SEC
02760           SBC TEMP
02770           STA ERROR
02780           LDA #0
02790           SBC TEMP+1
02800           STA ERROR+1
02810 ;
02820 ;
02830 ; FOR I.LOOP := 1 TO DELTA.X
02840 ;
02850 DO.LOOP
02860 ;
02870 ;
02880           LDA ERROR+1
02890           BMI ERROR.LT.0
02900 ;
02910 ;
02920 ; Y := Y NEG.Y 1
02930 ;
02940           LDY Y
02950           INY
02960           LDA NEG.Y
02970           BPL OK2
02980 ;
02990           DEY
03000           DEY
03010 ;
03020 OK2       STY Y
03030 ;
03040 ; ERROR :=ERROR+DELTA.Y-DELTA.X
03050 ;
03060           LDA ERROR
03070           SEC
03080           SBC DELTA.X
03090           STA ERROR
03100           LDA ERROR+1
03110           SBC DELTA.X+1
03120           STA ERROR+1
03130 ;
03140 ;
03150 ERROR.LT.0
03160 ;
03170 ; ERROR := ERROR + DELTA.Y
03180 ;
03190           LDA ERROR
03200           CLC
03210           ADC DELTA.Y
03220           STA ERROR
03230 ;
03240           LDA ERROR+1
03250           ADC #0
03260           STA ERROR+1
03270 ;
03280 ;
03290 DO.X
03300 ;
03310 ; X := X NEG.X 1
03320 ;
```

```
03330           LDA X
03340           LDY NEG.X
03350           BMI SUB4
03360 ;
03370           CLC
03380           ADC #1
03390           STA X
03400           LDA X+1
03410           ADC #0
03420 ;
03430           LDY #0
03440           BEQ STORE4
03450 ;
03460 SUB4      SEC
03470           SBC #1
03480           STA X
03490 ;
03500           LDA X+1
03510           SBC #0
03520 ;
03530 STORE4    STA X+1
03540 ;
03550           JSR PLOT.POINT
03560 ;
03570           INC I.LOOP
03580           BNE .1
03590           INC I.LOOP+1
03600 .1        LDA I.LOOP  ;two byte
03610           CMP DELTA.X ;compare
03620           BNE DO.LOOP
03630 ;
03640           LDA I.LOOP+1
03650           CMP DELTA.X+1
03660           BNE DO.LOOP
03670 ;
03680           RTS
03690 ;
03700 ;
03710 ;The plot function is stored on
03720 ; page 6 since it is non
03730 ;                   relocatable
03740 ;
03750 ; Now the plot function
03760 ;
03770 PLOT.ERROR JMP $A04D
03780 ;
03790 ;Simialr protection to drawto
03800 ;                   function
03810 ;
03820 PLOT      PLA
03830           CMP #2
03840           BNE PLOT.ERROR
03850 ;
03860           PLA
03870           STA X+1
03880           PLA
03890           STA X
03900           PLA
03910           BNE PLOT.ERROR
03920           PLA
03930           STA Y
03940           CLD
03950 ;
03960 PLOT.POINT
03970 ;
03980 ; divide x by 8 saving remainder
03990 ; Do not alter current value of x
04000 ; remainder is the bit offset
04010 ; quotient is the byte offset
04020 ;
04030 ; Offsets are from top left of
04040 ;                 screen
04050 ;
04060           LDA X       ;put x
04070           STA COPY    ;into work
04080           LDA X+1     ;space
04090           STA COPY+1
04100           LDA #0
04110           STA TEMP+1
04120           TAY         ;set y reg
04130 ;
04140 ;The hi byte of x only needs to
04150 ; be shifted once since only bit
04160 ; will ever be set
04170 ;
04180 ; This saves two instructions
04190 ;
04200 ;Move bits which 'fall out' into
04210 ; the accumulator as  remainder
04220           LSR COPY+1
04230           ROR COPY
04240           ROL
04250 ;
04260           LSR COPY
04270           ROL
04280 ;
04290           LSR COPY
04300           ROL
04310 ;
04320           TAX     ;put remainder
04330           LDA Y      into x
04340           STA TEMP
04350 ;
04360 ;
04370 ; multiply y by 40 because there
04380 ; are 40 bytes per screen line
04390 ;
04400 OVER      ASL TEMP
04410           ROL TEMP+1  ;times 2
04420           ASL TEMP
04430           ROL TEMP+1  ;times 4
04440           ASL TEMP
04450           ROL TEMP+1  ;times 8
04460 ;
04470           LDA TEMP    ;save times
04480           STA EIGHT   ;8
04490           LDA TEMP+1  ;for later
04500           STA EIGHT+1
04510 ;
04520           ASL TEMP
04530           ROL TEMP+1  ;times 16
04540           ASL TEMP
04550           ROL TEMP+1  ;times 32
04560 ;
04570           LDA TEMP
04580           CLC         ;add screen
04590           ADC SAVMSC  ;address to
04600           STA TEMP
04610           LDA TEMP+1  ;temp
04620           ADC SAVMSC+1
04630           STA TEMP+1
04640 ;
04650           LDA TEMP    ;add y
04660           CLC         ;times
04670           ADC EIGHT   ;8
04680           STA TEMP    ;from
04690           LDA TEMP+1  ;earlier
04700           ADC EIGHT+1
04710           STA TEMP+1
04720 ;
04730           LDA TEMP    ;add in
04740           CLC         ;x
04750           ADC COPY    ;divided
04760           STA TEMP    ;by 8
04770           LDA TEMP+1
04780           ADC COPY+1
04790           STA TEMP+1
04800 ;
04810           LDA COLOUR  ;if COLOR 0
04820           BEQ BLANK   ;  then
04830 ;               unplot the point
04840 ;
04850           LDA (TEMP),Y
04860           ORA PIXEL.PTR,X
04870           STA (TEMP),Y
04880           RTS         ;all done
04890 ;
04900 ;
04910 BLANK     LDA PIXEL.PTR,X get mask
04920           EOR #$FF    ;invert mask
04930           AND (TEMP),Y clear pixel
04940           STA (TEMP),Y ;to screen
04950           RTS         ;all done
04960 ;
04970 ;
04980 ;
04990 ; pixel.ptr converts  remainder
05000 ; into the mask for that pixel
05010 ;
05020 PIXEL.PTR .HS 8008200240041001
05030 ;
05040 ;
05050 ;
05060 ;General workspace
05070 ;
05080 EIGHT     .BS 2 ;8 times y
05090 COPY      .BS 2 ;a copy of x
05100 I.LOOP    .BS 2 ;loop counter
05110 NEG.X     .BS 1 ;sign of x inc.
05120 NEG.Y     .BS 1 ;sign of y inc.
05130 DRAW.X    .BS 2 ;x and y co-ords
05140 DRAW.Y    .BS 1 ;     to draw to
05150 DELTA.X   .BS 2 ;x increment
05160 DELTA.Y   .BS 1 ;y increment
05170 ERROR     .BS 2 ;two byte error
```

USER GROUP SOFTWARE

Software Librarian - Roy Smith

Due to demand from members there are now two ways to get programs from the library. The original scheme of exchanging '3 for 1' will still apply, but now with an added bonus. So the library rules have been extended to enable those members who cannot write their own programs to gain access, and those that can to have a possibility of some reward for their efforts. The extended library rules are as follows:

3 FOR 1 EXCHANGE

1. Every program you donate to the library entitles you to three programs in return.
2. The program you donate must be your original and not copied.
3. Your donated program must be submitted on a cassette or a disk, programs in the form of print-outs will not be processed.
4. If your program requires any special instructions they should be added in the form of REM statements within the program (or you may present them as instructions when the program is actually run).
5. BONUS. Every program donated per quarter (between issues of the newsletter) will be eligible to be judged 'STAR PROGRAM' for that quarter. This carries a prize of £10 which will be paid to the author from the club funds. The programs will be judged by the Editorial Team and their decision will be final. The Editorial Team are not eligible for the prize.
6. The '3 FOR 1' exchange is only open to club members.

DONATION SCHEME

1. Every club member will be entitled to ask for up to 3 programs per quarter from the library by donating to the club funds.
2. If a member does not take his/her entitlement for a particular quarter, it cannot be carried forward to the next quarter.
3. A member can have more than one quarter's entitlement at one time (up to a maximum of 12 programs (1 year)), but then will be unable to ask for more until his/her credit quarters have been used. Note that odd numbers of programs will be counted in quarters, i.e. if a member asks for 5 programs, the first 3 will be that quarter's entitlement, the next 2 will be the second quarter's entitlement and he/she will have to wait until the third quarter before he/she is entitiled to any more. Also note that having programs in advance will only be allowed if that member's membership covers the advance quarters.
4. The donation fee will be £1 per program and is not refundable. Cheques and Postal Orders are to be made out to the 'U.K. Atari Computer Owners Club'.
5. Members must send in a blank cassette or diskette for the chosen programs to be recorded on.
6. The 'DONATION SCHEME' is only open to club members.

Finally I would like to point out that some people omit to include return postage when donating to the library, so please do not forget to include 30p worth of stamps to cover this.

THE LIBRARY SOFTWARE SERVICE IS FOR MEMBERS ONLY

LIBRARY SOFTWARE TITLES

Listed below are the software titles received by members for inclusion in the library since issue six was published. As the library now contains over 300 programs it is getting a bit too large to keep on reprinting the entire list. Eventually it would probably take over the whole magazine and there would be no room left for the articles and program listings. For those of you who are new members and do not know what is available from the library then you can either purchase a back copy of issue six or send for a photocopy of the complete list which is available from the librarian. There is a small charge for this service to cover photocopying costs. If you would like a list please send 50p and a SAE for return.

Games

BULL ANT
by Sydney Brown – U.S.A.
Two player maze type game in which you must save your eggs from destruction.
Runs in 16K Cassette or 32K Disk min.
From ACE of Eugene, Oregon, U.S.A.

BUNNY RUN
by Steve Tullett – Dalkeith.
You are the rabbit and you need carrots to survive.
Runs in 16K min. Cassette only.

DOTMATCH
by Martin Bann – Basildon.
Guessing game, match the dots on the cards.
Runs in 16K Cassette or 32K Disk min.

FLASHING NUMBERS
by Chris Simeral – U.S.A.
Useful game for young children to make them concentrate.
Runs in any size Cassette or Disk.
From ACE of Eugene, Oregon, U.S.A.

FLIP 2
by Stephen Taylor – London.
Flip all the tiles in the correct sequence to win.
Runs in 32K Cassette or Disk min.

GUNFIGHT 2
by Grahame Fairall – Oxon.
Two player cowboy shootout.
Runs in 16K Cassette or Disk min.

JACKPOT
by J.P. Connell – Stoke-on-Trent.
Fruit Machine simulation including nudge and hold.
Runs in 16K min. Cassette only.
XL machines only.

KRAZY KLIMBER
by Sydney Brown – U.S.A.
Climb to the top of the building without getting knocked off!
Runs in 16K Cassette or 32K Disk min.
From ACE of Eugene, Oregon, U.S.A.

MATCHBOX
by Peter Cunningham – Chester.
Concentrate hard and match the shapes in the correct boxes.
Runs in 16K Cassette or 32K Disk min.

MILKY WAY
by Grahame Fairall – Oxon.
Pinball game created using Pinball Constructor.
Runs in 48K min. Disk only.

MUNCH
by Grahame Fairall – Oxon.
Two player game in which you eat as much as you can.
Runs in 16K Cassette or 32K Disk min.

PHAROAHS TOMB
by Sydney Brown – U.S.A.
Find the keys and unlock the door.
Runs in 16K Cassette or 32K Disk min.
From ACE of Eugene, Oregon, U.S.A.

QUICKTRACK
by Stephen Taylor – London.
Novel game of skill, guide the arrow to its target.
Runs in 32K Cassette or Disk min.
XL machines only.

RABBIT RUN
by Steve Tullett – Dalkeith.
Guide your bunny around the maze avoiding the Ferret.
Runs in 16K Cassette or Disk min.
Not XL compatible.

SNAPMATHS
by Steve Tullett – Dalkeith.
A snap game to help improve your maths, two player game.
Runs in 16K Cassette or Disk min.

SUPERFRUIT
by Ian Kinder – Stoke-on-Trent.
Slot machine game with nudge and hold.
Runs in 32K Cassette or Disk min.
XL machines only.

TIMEWORD
by Steve & Gwenda Tullett – Dalkeith.
Two player game based on scrabble.
Runs in 16K Cassette or Disk min.

Adventure Games

IDOL ISLAND
by J.P. Crackett – Choppington.
Extensive word only adventure the object of which is to search the island for treasure and live to tell the tale.
Runs in 48K Disk only.

LOST IN TIME
by M. Maestranzi – Finchley.
Word only adventure, can you complete the puzzles and win through?
Runs in 16K Cassette or 32K Disk min.

SIMPLE ADVENTURE
by Alex Kells – Liverpool.
Graphic adventure, find the hidden gem. Requires one complete side of a disk.
Runs in 32K min. Disk only.

THE CITADEL
by Steve Tullett – Dalkeith.
Rescue the Citadel from the brigands.
Runs in 32K Cassette or Disk min.

Home Entertainment

BIORHYTHM 2
by J. Bennett – Newcastle.
This program gives a graphic representation of your future well-being.
Runs in 16K Cassette or 32K Disk min.

CHANGE GIVER
by G. Berry – Wakefield.
Tells what coins to get in your change.
Runs in 16K Cassette or Disk min.

CHESS SET
by Steve Tullett – Dalkeith.
Chess set in GR.8 which can be used to play a game.
Runs in 16K Cassette or 32K Disk min.

TEXT GOLF
by Steve Tullett – Dalkeith.
Select the right club and shot to get round the course.
Runs in 16K Cassette or Disk min.

Demos

CHRISTMAS SCENE
by M. Maestranzi – Finchley.
Seasons greetings via computer graphics.
Runs in 16K Cassette or Disk min.

COLOR SELECTOR
by Ian Leonard – Chelmsford.
Use this program to find the colours you require.
Runs in 16K Cassette or Disk min.

FIREWORK
by Mark Christian – Wirral.
Randomly generated fireworks display with sound effects.
Runs in 16K Cassette or Disk min.

PICTURE SHOW 2
by Keith Berry – Birmingham
9 super pictures created using Fun with Art. One complete side of a disk is required.
Runs in 32K min. Disk only.

SHAPES 8
by Mark and Brian Christian – Wirral.
Interesting demo with some pretty nice shapes.
Runs in 16K Cassette or Disk min.

SINEWAVE
by Matthew Tydeman – Cheshunt.
Draw sinewave patterns on user input.
Runs in 16K Cassette or 32K Disk min.

VIDEO DISCS
by Keith Berry – Birmingham.
Graphics 8 demo with random elements showing video discs.
Runs in 48K min. Disk only.

Utilities

ANAGRAM 2
by Keith Berry – Birmingham.
Compiled version of earlier program which runs seven times faster.
Runs in any size Disk system.

ARTIST 2
by Martin Byfield – Birmingham.
Artistic sketch pad in GR.7+.
Runs in 32K Cassette or Disk min.

ARTIST SKETCH PAD
by David Harding – Manchester.
Draw pictures in GR.3,5,7 or 8.
Runs in 16K Cassette or 32K Disk min.

ASSEMBLER 2
by M.C. Barnard – Guisborough.
Improved version of Chris Rutter's assembler.
Runs in 32K min. Disk only.

BUBBLESORT
by M. Iremonger – Dublin.
Sorts a list and then dumps to a printer.
Runs in 16K Cassette or Disk min.

CASSETTE INSERT PRINTER
by Ian Leonard – Chelmsford.
Print out Insert Labels on an Epson MX80, includes cutting marks.
Runs in 16K min. Cassette only.

CHARACTER GENERATOR 5
by Stephen Bylo – Harpenden.
Design characters and save them to cassette.
Runs in 48K min. Cassette only.

CHOOSE COLOR
by Phil Brown – Newquay.
Select colours using this computer colour chart.
Runs in 16K Cassette or Disk min.

CONVERSION TABLE
by Mary Weightman – Bromham.
Converts distance, weight, area, speed and energy from/to imperial and metric.
Runs in 16K Cassette or Disk min.

DISK CONSISTENCY CHECKER
by Paul McAlinden – Airdrie.
Verifies that directory, bit map, etc. on a DOS 3 disk are intact.
Runs in 16K min. Disk only.

ERROR
by Stephen Taylor – London.
Gives written error messages in place of error numbers.
Runs in 16K Cassette or Disk min.

ERROR 2
by David Pink – Swanley.
Supplies error messages instead of code numbers.
Runs in 16K Cassette or Disk min.

GLOBAL CHANGE
by David Fuller – U.S.A.
Gives global change facility on Filemanager 800 files.
Runs in 16K Cassette or Disk min.
From ACE of Eugene, Oregon, U.S.A.

GRAPHY
by M. Iremonger – Dublin.
Graph sets of numbers using auto-scaling, includes print option.
Runs in 32K Cassette or Disk min.

TOP TEN

1	(10)	USERCOMP	TREVOR SKEGGS
2	(2)	SYNTHESISER	CHRIS PAYNE
3	(4)	THE VALLEY	STEVE CALKIN
4	(1)	FOLLY OF E. KKHANN	ALEX KELLS
5	(3)	STONEVILLE MANOR	NIGEL HASLOCK
6	(–)	DISK FILE MANAGER	DERECK DODSON
7	(9)	FRUIT MACHINE	MIKE NASH
8	(–)	LABEL PRINTER	PETER BLACKMORE
9	(–)	DOGFIGHT	ROD KNOWLES
10	(–)	TRAPDOOR	BOB ASKEW

HOME BANK MANAGER
by Tony Grew – Birmingham.
Keep your finances upto date.
Runs in any size. Cassette only.

HOME FINANCE
by K. Vaughan – Basingstoke.
Keep a record of your bills.
Runs in any size. Cassette only.

JOHNNY'S PAINTBOX
by Stan Ockers – U.S.A.
Excellent picture painting program.
Runs in 48K min. Disk only.
ACTION! program (but ACTION! cartridge is not required).
From ACE of Eugene, Oregon, U.S.A.

LISTALL
by J.P. Crackett – Choppington.
Lists any ATASCII file to Epson printer (tested on RX80 F/T), includes inverse and graphics characters. Atari Macro Assembler source code.
Runs in 32K min. Disk only.

★★★★ STAR PROGRAM ★★★★

MASTERS DOS
by Matthew Tydeman – Cheshunt.
DOS system which includes its own character font.
Runs in any size. Disk only.

MEMORY DUMP
by Jeff Davies – Llandeilo.
Memory dump routine to the screen.
Runs in 16K Cassette or Disk min.

MC MONITOR
by Paul McAlinden – Airdrie.
Simple m/c monitor which permits display, modification, etc. of blocks of memory using Hex addresses and data.
Runs in 32K min. Disk only.

MENU
by M. Iremonger – Dublin.
Lists programs on disk, files can be locked, deleted, etc.
Runs in any size. Disk only.

MENU MAKER
by Mark & Brian Christian – Wirral.
Add this program to your disks to give a menu of files.
Runs in any size. Disk only.

MENUMASTER
by Trevor Skeggs – Milton Keynes.
This Master Menu module gives colourful and customisable DLI's with control from joystick or keyboard.
Runs in 16K Cassette or Disk min.

MINIDOS 2
by Linda Tinkler – Wirral.
All the usual disk operating options directly from BASIC.
Runs in any size. Disk only.

OLDTEXT
by Martin Byfield – Birmingham.
Change Atari text into old fashioned lettering.
Runs in 32K Cassette or Disk min.

PMG COLOURS
by S. Richards – Acomb.
Select colours for your PMG characters.
Runs in any size. Cassette only.

REMOVE
by Jeff Davies – Llandeilo.
Deletes lines using forced read mode.
Runs in 16K Cassette or Disk min.

ROUTINE NOTE
by J.P. Connell & Ian Kinder – Stoke-on-Trent.
A little routine which plays notes according to user input.
Runs in 16K Cassette or Disk min.

SCROLLING MESSAGE
by M. Maestranzi – Finchley.
BASIC program displays message at the top of the screen, upto 220 characters can be used.
Runs in 16K Cassette or Disk min.

SOUND SYSTEM
by Matthew Tydeman – Cheshunt.
Experiment with the sound on your Atari. Requires joystick and Audio cassette.
Runs in 32K Cassette or Disk min.

TEXT FILE MANAGER
by Ian Scott – Tyne & Wear.
Disk based BASIC program to create and edit text files.
Runs in any size. Disk only.

VATCALC
by Ian Leonard – Chelmsford.
Add and subtract VAT from figures input by user.
Runs in 16K Cassette or 32K Disk min.

WORLD CLOCK
by David Pink – Swanley.
Gives list of times around the world.
Runs in 16K Cassette or Disk min.

YEARLY BARGRAPH
by M. Iremonger – Dublin.
Graphs monthly income upto 2 years.
Runs in 16K Cassette or 32K Disk min.

Education

ALPHABET BEE
by Stan Ockers – U.S.A.
Childrens teaching aid.
Runs in 48K min. Disk only.
ACTION! program (but ACTION! cartridge is not required).
From ACE of Eugene, Oregon, U.S.A.

BOOKS OF THE BIBLE
by Gwenda Tullett – Dalkeith.
A quiz about the books of the Bible.
Runs in 16K Cassette or Disk min.

ELECTRONIC DICTIONARY
by Paolo Fragapane – Bristol.
Create your own foreign language dictionary by using the forced read mode to input data.
Runs in any size. Cassette only.

MATHS PROBLEMS
by Brian Christian – Wirral.
3 programs to solve Simultaneous Equations, Indices and Triangles.
Runs in 32K Cassette or Disk min.

Music

AWAY IN A MANGER
by Mary Weightman – Bromham.
The lovely xmas carol with verses to sing, runs from BASIC.
Runs in 16K Cassette or Disk min.

MUSIC 3
by Grahame Fairall – Oxon.
6 tunes: Isle of Capri, Hawaii 5-0, Match of the Day, Whiter Shade of Pale, Lion Sleeps Tonight, Steptoe & Son. Atari Music Composer cartridge required.
Runs in 16K Cassette or Disk min.

MUSIC 4
by Grahame Fairall – Oxon.
6 tunes: Cinderella Rockafella, Diamonds are a Girls Best Friend, Tie a Yellow Ribbon, Star Wars, Milord, Organ Waltz. Atari Music Composer cartridge required.
Runs in 16K Cassette or Disk min.

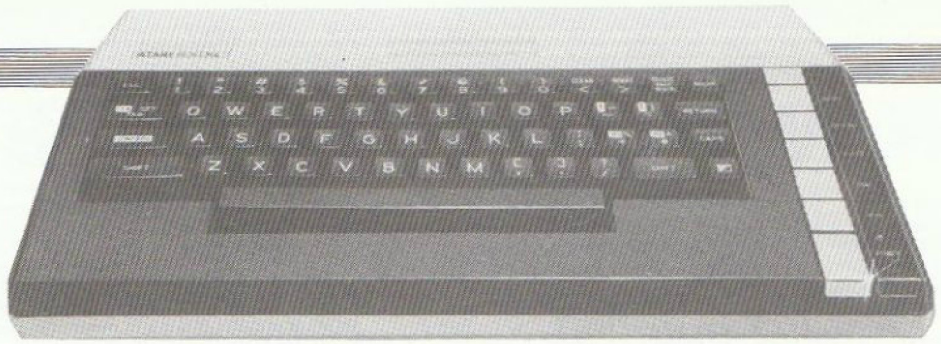
THREE TUNES
by Grahame Fairall – Oxon.
Run Rabbit, Entertainer and Camptown Races, in Basic.
Runs in 16K Cassette or 32K Disk min.

CORRIGENDA ISSUE 7

FLIP
Add on the end of line 301, CX=0. Delete line 2095. Line 1020 should read IF NT(0,0)=G OR NT(1,0)=G OR NT(0,1)=G OR NT(1,1)=G THEN GOTO 1036.

LANDSCAPE
Unfortunatly false information was supplied to us about this program, it is in fact 'Stealth' by Broderbund.

ATTENTION ATTENTION
ATARI 400/600/800 OWNERS
MIDLAND GAMES LIBRARY
Do you want to join a long established library?
Are you looking for a fast efficient and friendly service?
Would you like to select from nearly 850 programs; cassettes, cartridges, discs and utilities and educational?
Would you appreciate 40 new additions per month?
Are you interested in interactive club schemes?
Two games may be hired at any one time.
We buy many of the popular games in multiples of five or six to give all our members a fair choice.
Now entering our third year of service to Atari owners.
Hundreds of satisfied members, many even as far away as Iceland, Denmark, Eire and West Germany.
Send large SAE for details.
M.G.L. (M8)
48 Read Way, Bishops Cleeve, Cheltenham
(0242-67) 4960 6pm-9pm
All our games are originals with full documentation



INTERRUPTS

BY STEVE HILLEN Part 2

Fine Scrolling

The last use for interrupts (at least for this article), is to achieve flicker-free fine scrolling. As you are probably already aware, fine scrolling on the Atari is made very easy by two features: the display list, and the hardware scrolling registers, HSCROL (horizontal scroll 54276), and VSCROL (vertical scroll 54277).

Let's start this section with vertical scrolling. Suppose we call up GRAPHICS 8+16 through BASIC. We'll see 192 lines each displaying 40 bytes of consecutive memory, starting with the address given by the 5th and 6th bytes of the display list. If we alter these two bytes we'll alter the area of memory being shown. Try:

```
DL=PEEK(560)+256*PEEK(561)
POKE DL+4,PEEK (DL+4)-40
```

By subtracting 40 from the operand byte we effectively shift the top half of the TV screen down. Since each GRAPHICS 8 mode line is in fact one scan line deep, we have shifted the display down by one scan line, and hence have fine-scrolled the display. Note that since the GRAPHICS 8 screen needs more than 4K of RAM, there is a second LMS (Load Memory Scan) instruction which would also have to be altered if the whole screen were to be scrolled. This same method of vertical scrolling can be used for all graphics modes, but with modes that are, say, 8 scan lines high e.g. GR.0, each scrolling operation will move the display by 8 lines, which produces a very coarse effect.

To avoid this problem we must use a different method of scrolling. This involves modifying the display list and poking various values into VSCROL. The modification necessary is to "enable" vertical fine scrolling by adding 32 to the mode line byte that you wish to be scrolled. This is illustrated in the example display list below:

Instruction	Normal	Modified
Blank Line	112	112
Blank Line	112	112
Blank Line	112	112
LMS	66	98
Area to	00	00
Display	144	144
GR.0	2	34
GR.0	2	34
GR.0	2	34
Last line	2	2
JVB	65	65

Note that this display list has a normal non-scrolling last line. This is necessary because the last line of a vertically scrolling screen is not shown, it is kept as a buffer containing the data that is about to be displayed.

Now we can shift the display up and down by any number of scan lines just by poking this number into VSCROL. The appropriate ranges of the acceptable values to poke are listed in table 4. This permits fine scrolling through character and low resolution graphics modes. Once either the maximum or minimum values as given by table 4 are reached, then they should wrap round to the other limit. Also at this time, the whole display must be coarse-scrolled one mode line by adding or subtracting 40 bytes from the LMS operand bytes. This produces the effect of continuous fine scrolling, see diagram 3.

BASIC Graphics Mode	VSCROL Range	HSCROL Range
0	0–7	0–3
1	0–7	0–7
2	0–15	0–7
3	0–7	0–15
4	0–3	0–15
5	0–3	0–7
6	0–1	0–7
6+(12)	0**	0–7
7	0–1	0–3
7+(14)	0**	0–3
8	0**	0–3
9,10,11	0**	0–3*

*Having HSCROL set to odd numbers causes GTIA to be turned off so use only 0 and 2.
**Just update LMS operands.

Table 4.

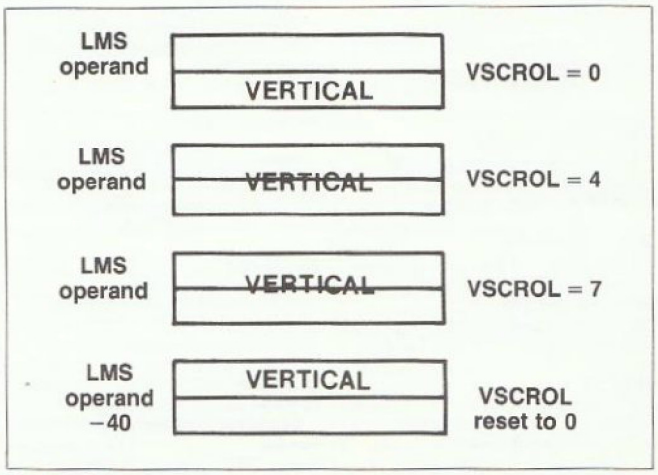


Diagram 3.

Listing 7.

```
 5 REM xxxM/c data followsxxx
 10 DATA 112,112,66,35,56,98,0,0,34,34,34,34,34,34,34,34,
 34,34,34,34,34,34,34,34,34,34,34,34
 20 DATA 34,34,34,2,65,0,56,54,101,114,116,105,99,97,108,
 0,51,99,114,111,108,108,101,114,0,4,16,16
 30 DATA 16,16,0,116,111,0,4,16,16,16,16,104,169,33,141,4
 7,2,162,0,142,6,56,142,7,56,142,5,212
 40 DATA 142,255,6,169,0,141,48,2,169,56,141,49,2,32,210,
 56,169,6,162,56,160,109,76,92,228,32,118,56
 50 DATA 32,210,56,76,95,228,173,0,211,41,1,208,41,174,25
5,6,232,224,8,208,26,173,7,56,201,253,240,25
 60 DATA 173,6,56,24,105,32,141,6,56,173,7,56,105,0,141,7
,56,162,0,142,255,6,142,5,212,96,173,0
 70 DATA 211,41,2,208,248,174,255,6,202,16,236,173,7,56,2
08,5,173,6,56,240,232,173,6,56,56,233,32,141
 80 DATA 6,56,173,7,56,233,0,141,7,56,162,7,208,205,173,6
,56,32,3,57,141,57,56,141,66,56,142,56
 90 DATA 56,142,56,56,142,65,56,173,7,56,32,3,57,141,55,5
6,142,54,56,173,7,56,24,105,3,32,3,57
 100 DATA 141,64,56,142,63,56,96,72,74,74,74,74,170,189,2
1,57,170,104,41,15,168,185,21,57,96,16,17,18
 110 DATA 19,20,21,22,23,24,25,33,34,35,36,37,38
 120 REM xxxInstall m/c on page 56xxx
 130 FOR A=0 TO 297:READ D:POKE 56*256+A,D:NEXT A
 140 SETCOLOR 2,0,0:REM xxxInit. colourxxx
 150 REM xxxCall routinexxx
 160 X=USR(56*256+67)
```

Listing 8.

```
00010 ;Vertical memory scroll
00020 ;by S.Hillen
00030 ;
00040          .LI OFF
00050 ;Equates
00060 ;
00070 SDMCTL   .EQ $22F    DMA c'tl
00080 SDLSTL   .EQ $230    Dlist ptr
00090 SETVBV   .EQ $E45C   Set VBI
00100 SYSVBV   .EQ $E45F   Jump OS
00110 VSCROL   .EQ $D405
00120 PORTA    .EQ $D300   Joystick
00130 S.VSCROL .EQ $6FF    Shadow
00140 SCREEN   .EQ 0
00150 ;
00160          .OR $3000
00170 ;
00180 DLIST    .HS 707042  Blank line
00190          .DA STATUS
00200          .HS 62      LMS+Scroll
00210 LMSLO    .DA SCREEN
00220          .HS 2222222222222222
00230          .HS 2222222222222222
00240          .HS 2222222222222202
00250          .HS 41      JVB
00260          .DA DLIST
00261 ;
00262 ;Text that appears on top line
00270 ;
00280 STATUS   .AT "Vertical Scroller $"
00290 LO       .AT "0000 to $"
00300 HI       .AT "0000"
00310 ;
00320 ;
00330 START    PLA         Discard
00340          LDA #33     Thin
00350          STA SDMCTL  screen
00360          LDX #0      Zero the
00370          STX LMSLO   LMS bytes
00380          STX LMSLO+1 and also
00390          STX VSCROL  scrolling
00400          STX S.VSCROL registers
00410          LDA #DLIST  Point to
00420          STA SDLSTL  our new
00430          LDA /DLIST  display
00440          STA SDLSTL+1 list then
00450          JSR INFOLINE show area
00460          LDA #6      Immediate
00470          LDX /VBI    VBI
00480          LDY #VBI    & let OS
00490          JMP SETVBV  carry on.
00500 ;
00510 VBI      JSR JOYSTICK
00520          JSR INFOLINE
00530          JMP SYSVBV  Into OS.
00540 ;
00550 JOYSTICK
00560          LDA PORTA   Joystick
00570          AND #1      Up?
00580          BNE NOT.UP  No
00590          LDX S.VSCROL Is VSCROL
00600          INX         too large?
00610          CPX #8
00620          BNE DO      No
00630          LDA LMSLO+1 It is,so
00640          CMP #$FD    time to
00650          BEQ OUT     alter the
00660          LDA LMSLO   LMS bytes,
00670          CLC         if they
00680          ADC #32     aren't too
00690          STA LMSLO   large!
00700          LDA LMSLO+1 16 bit add
00710          ADC #0      of 32 ie.
00720          STA LMSLO+1 one line
00730          LDX #0      extra.
00740 DO       STX S.VSCROL Reset it
00750          STX VSCROL  & hardware
00760 OUT      RTS         Return.
00770 NOT.UP   LDA PORTA   Joystick
00780          AND #2      Down?
00790          BNE OUT     No
00800          LDX S.VSCROL Yes,so
00810          DEX         check if
00820          BPL DO      VSCROL is
00830          LDA LMSLO+1 too small.
00840          BNE .2      Alter LMS
00850          LDA LMSLO   if not too
00860          BEQ OUT     small.
00870 .2       LDA LMSLO   A 16 bit
00880          SEC         subtract
00890          SBC #32     to shift
00900          STA LMSLO   screen up
```

```
00910          LDA LMSLO+1 by one
00920          SBC #0      line.
00930          STA LMSLO+1
00940          LDX #7
00950          BNE DO
00980 ;
00990 INFOLINE
01000          LDA LMSLO   Now show
01010          JSR SPLIT   the area
01020          STA LO+3    of memory
01030          STA HI+3    in the
01040          STX LO+2    display.
01050          STX LO+2
01060          STX HI+2
01070          LDA LMSLO+1
01090          JSR SPLIT
01100          STA LO+1
01120          STX LO
01130          LDA LMSLO+1 The upper
01140          CLC         limit is
01150          ADC #3      always 3
01160          JSR SPLIT   pages
01170          STA HI+1    higher
01190          STX HI      than the
01200          RTS         lower.
01210 ;
01220 SPLIT
01230          PHA         Save hi
01240          LSR         Move rh.
01250          LSR         nibble
01260          LSR         into lh.
01270          LSR         nibble
01280          TAX         and get
01290          LDA TABLE,X Atascii
01300          TAX         byte from
01310          PLA         table.
01320          AND #$0F    Do same
01330          TAY         for lh.
01340          LDA TABLE,Y nibble.
01350          RTS
01360 ;
01370 TABLE    .AT "0123456789ABCDEF"
```

If the above sequence is synchronised to the TV display by interrupts, then flicker-free fine scrolling is produced as in listing 7. (Don't forget to save it before running it). This program runs in VBI and is controlled by the joystick in port 1. It is a vertical scroll demo, that runs over the entire 64K memory range of the computer, displaying 768 bytes (3 pages) per screen. There are a number of features of this program that should be noted:

1. The last mode line is not vertical scroll enabled to produce a steady last line. Try poking 14367,34 to see what happens without it.
2. The screen has been made narrow, this was to simplify the programming.
3. The screen flicks as you cross a 4K boundary ($1000, $2000 . . . etc). This is because the hardware was not designed to cross a 4K boundary without another LMS instruction.
4. The sequence of operations performed in VBI is as follows: Joystick is read, VSCROL is adjusted (VSCROL is a write-only register, thus a shadow is kept). If VSCROL reaches a limit (0 or 7), then the LMS operands are adjusted by 32 bytes (narrow screen only uses 32 bytes per line).
5. Try looking at page 0: the changing characters on the first line are the internal clock. If you type LIST you will see BASIC adjusting pages 0 and 1.
6. If you go far enough up in memory you will encounter the normal display screen that should contain your last command.
7. If you try to display regions of memory that are not present in your computer expect some flicker.
8. This program does not work with K-DOS as it was written to be compatible with 16K machines.
9. The source code is in listing 8.

By using this technique of having the TV screen as a "viewing window" on a far larger memory area games such as Caverns of Mars are created.

Horizontal Scrolling

Horizontal scrolling is achieved in a very similar manner as vertical scrolling. This time, the display list is adjusted by adding 16 to the mode lines that you wish to scroll, or 48 if both forms of scrolling are desired. Also, each line that is to be scrolled must have its own separate LMS instruction. If many lines are to be horizontally scrolled, a fairly well-organised memory map is necessary. One point to remember in organising the memory layout is that a standard width screen (40 bytes GR.0), will need

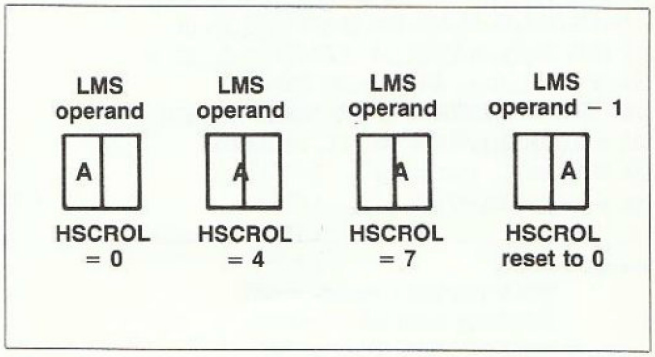


Diagram 4.

48 bytes if it is to be horizontally scrolled. This is similar to the buffer used when vertically scrolling.

This time you must poke HSCROL with the appropriate range of values (table 4). Once either limit is reached, on the next operation you should wrap round this value to the other limit and also add or subtract 1 to the LMS operand bytes. This effectively smooth scrolls continuously (see diagram 4).. Note again that HSCROL is also a write-only register (shadow needed!).

The demo program for horizontal scrolling is, for a change, running as a DLI, and will horizontally scroll any one mode line in either direction at any speed. There are a number of modifications that are necessary to your display list in order to run this program.

1. Add 128 to the first 112 (8 blank lines) to allow this DLI to occur.
2. Adjust your display list so that the scrolled line has its own LMS + horizontal scroll enable (64+16+mode). Make a note of this number.
3. Decide where to display memory from (in this case I chose 'buffer' to be on page 57) and install your LMS operand bytes into the display list such that they lie within the range of the region you want scrolled.
4. Point to your new display list with pokes to 560 and 561.
5. Call my routine with: X=USR(ADR(SC$),HSCROL MAX, START,END,SPEED,DIRECTION,BYTE,ADR(SC$)+80). Where HSCROL MAX is the maximum value for your graphics mode from table 4. BYTE is the LMS opcode that you noted in (2). The program locates the LMS opcode from this. START,END are the starting and ending addresses of the buffer to be scrolled. Note that if you want a continuously scrolling message as in the demo, then the

scrolled region must continue for at least 40 bytes further than END, otherwise a blank region will scroll in, followed by the starting display suddenly flashing in. DIRECTION 0 = right, 1 = left. SPEED, anything up to the maximum of HSCROL range for that graphics mode. 0 = no movement.

The source code is in listing 10. By writing other routines that control many horizontal scrolling lines, you can create 'SCRAMBLE' type games or scrolling 80/120 column word-processors etc. Unfortunately, the last two programs are rather specialised, but scrolling has such a wide range of applications it would be impossible to provide routines for general use. However, I hope that this article has shown you how easy it is to use interrupts to improve graphics in your programs and you should feel free to incorporate any of the routines in this article in your own programs.

In this program, anything which is underlined, should be entered in 'INVERSE'.

Listing 9.

```
5 REM xxxSet up M/c data in SC$xxx
10 DIM SC$(255):FOR A=1 TO 255:READ D:SC$(A)=CHR$(D):NEX
T A
20 DATA 104,104,104,141,0,6,104,141,1,6,104,141,2,6,104,
141,3,6,104,141,4,6,104,104,141,5,6,104
30 DATA 104,141,6,6,173,49,2,133,204,173,48,2,133,203,16
0,0,140,9,6,140,4,212,104,104,141,10,6,177
40 DATA 203,200,205,10,6,208,248,140,7,6,104,141,1,2,104
,141,0,2,169,192,141,14,212,96,120,72,152,72
50 DATA 138,72,173,8,6,240,89,216,173,48,2,133,203,173,4
9,2,133,204,173,9,6,174,6,6,240,77,56,237
60 DATA 5,6,16,55,24,109,0,6,170,232,172,7,6,177,203,200
,205,4,6,208,22,177,203,205,3,6,208,15
70 DATA 173,1,6,145,203,136,173,2,6,145,203,138,56,176,1
6,136,177,203,24,105,1,145,203,200,177,203,105,0
80 DATA 145,203,138,141,10,212,141,4,212,141,9,6,104,170
,104,168,104,88,64,24,109,5,6,205,0,6,144,231
90 DATA 240,229,237,0,6,170,202,172,7,6,177,203,200,205,
2,6,208,22,177,203,205,1,6,208,15,173,3,6
100 DATA 145,203,173,4,6,136,145,203,138,24,144,191,136,
177,203,56,233,1,145,203,200,177,203,233,0,145,203,138
110 DATA 56,176,172
115 REM xxxString to be scrolledxxx
116 REM xxxNB. repeated section at endxxx
119 DIM TEXT$(120)
120 TEXT$="NOW YOU'VE READ THIS ARTICLE,TRY SOMETHING EA
SIER-THE ADVENTURE COLUMN. NOW YOU'VE READ THIS ARTI"
125 REM xxxNew display listxxx
126 REM xxxFirst line has interruptxxx
127 REM xxxScrolled line has its ownxxx
128 REM xxxLMS byte...xxx
130 FOR A=0 TO 23:READ D:POKE 1664+A,D:NEXT A
140 DATA 240,112,112,71,0,58,7,7,7,7,87,3,57,71,0,58,7,7
,7,7,7,65,128,6
150 REM xxxCopy TEXT$ into our bufferxxx
155 FOR A=1 TO LEN(TEXT$):POKE 57*256+3+A,ASC(TEXT$(A,A)
)-32:NEXT A
160 ? "INPUT SPEED(1 TO 7)":INPUT SPEED:IF SPEED>7 OR SP
EED<1 THEN 160
165 ? "INPUT DIRECTION 0=RIGHT,1=LEFT":INPUT DIRECTION:I
F DIRECTION>1 OR DIRECTION<0 THEN 165
185 POKE 560,128:POKE 561,6:REM xxx Point to display lis
txxx
190 X=USR(ADR(SC$),7,57*256+3,57*256+74,SPEED,DIRECTION,
87,ADR(SC$)+80)
195 POKE 1544,1:REM xxxEnable itxxx
```

Listing 10.

```
00010 ;Utility to horizontally
00020 ;scroll 1 mode line.
00030 ;by S.Hillen
00040 ;
00050          .LI OFF
00060 ;Equates
00070 ;
00080 VDSLST   .EQ $200    DLI vector
00090 SDLSTL   .EQ $230    Dlist ptr.
00100 HSCROL   .EQ $D404
00110 NMIEN    .EQ $D40E
00120 WSYNC    .EQ $D40A
00130 ;
00140          .OR $600
00150 ;
00160 MODEMAX  .BS 1       HSCROL max
00170 STARTHI  .BS 1       Start of
00180 STARTLO  .BS 1       buffer.
00190 ENDHI    .BS 1       End of
00200 ENDLO    .BS 1       buffer.
00210 SPEED    .BS 1
00220 DIRECTION .BS 1
00230 POINTER  .BS 1       Dlist ptr.
00240 ENABLE   .BS 1
00250 S.HSCROL .BS 1       Shadow loc
00260 BYTE     .BS 1       LMS byte
00270 DLIST    .EQ $CB     Page 0
00280 ;
00290          .OR $4000   Anywhere
00300 ;
00310          PLA         Discard
00320          PLA         Discard hi
00330          PLA         Lo is max
00340          STA MODEMAX HSCROL val
00350          PLA         Buffer
00360          STA STARTHI start
00370          PLA         address.
00380          STA STARTLO
00390          PLA         Save the
00400          STA ENDHI   buffer
00410          PLA         end
00420          STA ENDLO   address.
00430          PLA         Discard hi
00440          PLA         Save the
00450          STA SPEED   speed.
00460          PLA         Discard hi
00470          PLA         Save dire-
00480          STA DIRECTION  ction
00490          LDA SDLSTL+1 Set up
00500          STA DLIST+1 page 0 ptr
00510          LDA SDLSTL  to the
00520          STA DLIST   disp. list
00530          LDY #0      Zero out
00540          STY S.HSCROL scrolling
00550          STY HSCROL  locations.
00560          PLA         Discard hi
00570          PLA         Get the
00580          STA BYTE    LMS byte
00590 SCAN     LDA (DLIST),Y  and
00600          INY         scan Dlist
00610          CMP BYTE    for it.
```

```
00620          BNE SCAN     Keep Y as
00630          STY POINTER an index.
00640          PLA          Address of
00650          STA VDSLST+1 our DLI
00660          PLA          stored in
00670          STA VDSLST   DLI vector
00680          LDA #$C0     Enable
00690          STA NMIEN    DLI's and
00700          RTS          return.
00710 ;
00720 ;
00730 DLI
00740          SEI          Prevent
00750          PHA          further
00760          TYA          interrupts
00770          PHA          and save
00780          TXA          registers
00790          PHA          on stack.
00800          LDA ENABLE   Enabled?
00810          BEQ OUT      No
00820 ;
00830 SCROLL
00840          CLD          Binary
00850          LDA SDLSTL   Set up
00860          STA DLIST    page 0
00870          LDA SDLSTL+1 pointer
00880          STA DLIST+1  for later.
00890          LDA S.HSCROL
00900          LDX DIRECTION  Which
00910          BEQ RIGHT      way?
00920 LEFT
00930          SEC          Alter
00940          SBC SPEED    S.hscrol
00950          BPL LEAVE    with a
00960          CLC          wraparound
00970          ADC MODEMAX  if <0
00980          TAX          Save in X
00990          INX          Add one.
01000          LDY POINTER  Check that
01010          LDA (DLIST),Y  LMS not
01020          INY          too high.
01030          CMP ENDLO    If it is
01040          BNE .1       then need
01050          LDA (DLIST),Y to reset
01060          CMP ENDHI    the LMS
01070          BNE .1       bytes to
01080 ;
01090          LDA STARTHI  their
01100          STA (DLIST),Y starting
01110          DEY          values.
01120          LDA STARTLO
01130          STA (DLIST),Y
01140          TXA          Restore
01150          SEC          new HSCROL
01160          BCS LEAVE    Always!
01170 ;
01180 .1       DEY          Just inc.
01190          LDA (DLIST),Y  the lo
01200          CLC          of the LMS
01210          ADC #1       operand
01220          STA (DLIST),Y  byte,
01230          INY          by 16 bit
01240          LDA (DLIST),Y
01250          ADC #0       addition.
01260          STA (DLIST),Y
01270 .2       TXA          Restore
01280 ;                     HSCROL &
01290 LEAVE
01300          STA WSYNC    Wait,
01310          STA HSCROL   Save in
01320          STA S.HSCROL   both
01330 ;
01340 OUT
01350          PLA          Restore
01360          TAX          all
01370          PLA          registers
01380          TAY
01390          PLA          Re-enable
01400          CLI          interrupts
01410          RTI          & return
01420 ;
01430 RIGHT
01440          CLC
01450          ADC SPEED    Increase
01460          CMP MODEMAX HSCROL,
01470          BCC LEAVE    with
01480          BEQ LEAVE    wraround
01490          SBC MODEMAX if too
01500          TAX          large,
01510          DEX          Save in X
01520          LDY POINTER  Now check
01530          LDA (DLIST),Y  that
01540          INY          operands
01550          CMP STARTLO  aren't too
01560          BNE .1       small.If
01570          LDA (DLIST),Y they
01580          CMP STARTHI  are, then
01590          BNE .1       reset them
01600 ;
01610          LDA ENDHI    to the end
01620          STA (DLIST),Y thus.
01630          LDA ENDLO
01640          DEY
01650          STA (DLIST),Y
01660          TXA          Restore
01670          CLC          HSCROL &
01680          BCC LEAVE    leave DLI.
01690 ;
01700 .1       DEY          Just dec.
01710          LDA (DLIST),Y  LMS
01720          SEC          operands
01730          SBC #1       by using
01740          STA (DLIST),Y  16 bit
01750          INY          subtraction
01760          LDA (DLIST),Y
01770          SBC #0
01780          STA (DLIST),Y
01790 .2       TXA          Restore
01800          SEC          HSCROL
01810          BCS LEAVE    and leave.
```

SPECIAL OFFERS

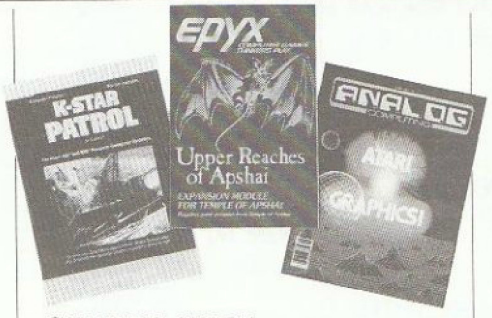
Several exciting pieces of software are on offer this quarter, but stocks are limited so please order promptly to avoid disappointment. Cheques or Postal Orders are to be made payable to the "U.K. Atari Computer Owners Club" and the prices quoted are valid until the end of May 1985. Please state cassette or disk where applicable.

K-STAR PATROL

While on patrol over an alien planet your team of Star Ships is attacked. Only the leading Star Ship can protect the squadron, so you must blast your way through the alien swarms outmanoeuvring the low-level avoidance system from each sector into the next, until you confront the evil energy absorbing leech.
ROM cartridge for 400 and 800 only.
Not XL compatible.
Usual Price £14.95 **Club Price £7.00**

SHOOTING ARCADE

Excellent fairground shooting gallery game which sports rabbits, ducks, balloons and funny faces. Also there is a bonus fatman to shoot at the end of each round.
16K Cassette or 32K Disk for 400, 800 or XL machines.
Usual Price £13.50 **Club Price £5.00**



GALACTIC CHASE

The aliens have swept undefeated across the galaxy and only you in your StarCraft can stop them. Watch out for the Flanker ships which swoop down on your position. This is an exciting game similar to Galaxians. It is for 1 or 2 players who select one of two games.
Runs in any size. Only available on disk for 400, 800 or XL machines.
Usual Price £7.50 **Club Price £5.00**

RICOCHET

This game is sub-titled a strategy game with bounce, and thats because it combines the fast action bouncing ricochet of a ball game with the subtle strategies of a board game. There are 5 game variants, 4 levels of skill and is for 1 or 2 players.
16K Cassette or 32K Disk for 400, 800 or XL machines.
Usual Price £7.95 **Club Price £5.00**

TEMPLE OF APSHAI (SET)

The complete set of Apshai 'Dunjon-quest' games. Explore the world of monsters and magic in this fantasy role playing experience of heroic adventure. The set comprises Temple of Apshai, Upper Reaches of Apshai & Curse of Ra.
32K Disk only for 400, 800 or XL machines.
Usual Price £37.35 (Set of 3)
Club Price £14.00 (Set of 3)

ANALOG MAGAZINE

Issues 16, 17 & 18 are available and contain many interesting and informative articles, so if you are missing these issues from your collection or have never seen a copy of Analog, then purchase a copy of these back issues to see what you have missed. Incidentally Maplin Electronics are still offering a regular subscription to Analog at £18.00 for six issues (their code number is GG24B).
The special offer price for issues 16, 17 & 18 includes P & P. **Club Price £1.90 each.**

MOONBASE PLATO™

It is now 2023, almost 20 years since the first permanent Lunar Base was set up. Since then, many new industrial bases have been built, the companies taking advantage of the low gravity to pioneer new technologies. However, competition between the larger companies increased until some resorted to theft in order to stay in the forefront.

As Security Officer to LunMetal Ltd., one of the most advanced companies on the Moon, you thought this could never happen to you. Until it did. The technical documents for a revolutionary new reactor were stolen!

Realizing that the crime could only have been perpetrated by LunFuel Ltd., you set off in pursuit of the plans towards LunFuel's complex – Moonbase Plato. Flying high above the crater in the EVA pod, you witness a startling scene, the Moonbase is gutted and wrecked. The ground far below ripples convulsively as the last of the Lunar tremors die away.

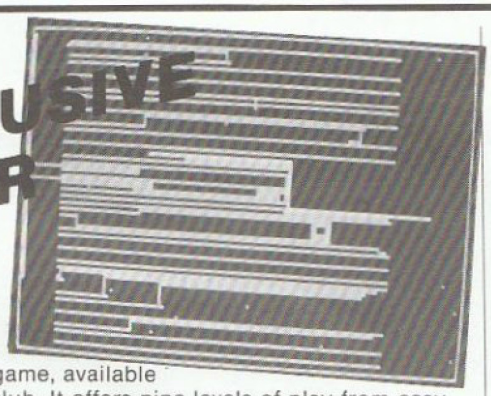
Nervously, you descend towards the crater floor, landing rather heavily on the one undamaged pad. Now the adventure begins.....

Moonbase Plato is an adventure that understands over 100 words, has a fast response time, room descriptions, and a split-screen display. It is available only from the User Group and will run on any machine with at least 32K of memory. There is of course a save game feature.

Send a blank Cassette or Disk and we will record an Auto-boot copy of MOONBASE PLATO onto it for you (a good quality C60 min. please). Make out a cheque/P.O. for £4.95 to 'The U.K. Atari Computer Owners Club'. The price includes postage and packing in U.K. and Eire, Overseas members should add 50p (Europe) or £1.00 (Outside Europe).

Please mark your parcel 'MOONBASE PLATO'.

EXCLUSIVE OFFER



TRAP is a new, 100% machine code game, available only from this club. It offers nine levels of play from easy to impossible and is for one or two players. You and the 'enemy' must fill in all the open spaces with your trailing tails, avoiding the mines left in your way. If you out last all of the enemies then you are awarded a point for every enemy ranged against you on that screen, but if only one enemy out lives you then you get nothing. In the two player option, if all the enemies are destroyed then the player who lasts the longest gets the points. For every 100 points you collect you are given an extra life.

TRAP is obtainable by sending a cheque/postal order for £3.95, made out to the club, to P.O. Box 3, Rayleigh, Essex together with a blank cassette or disk. An auto-boot copy of TRAP will be recorded onto your cassette or disk and then returned to you. By marketing TRAP along similar lines to the Library Donation Scheme the overall costs can be kept low and many more people will feel the benefit. The price includes postage (U.K. & Eire) and packing but Overseas members should add 50p (Europe) or £1.00 (Outside Europe). Please mark your package 'TRAP OFFER' and remember it is best to send a good quality cassette or disk and also ensure adequate wrapping is provided.

PAGE 6 THE MAGAZINE FOR ALL ATARI COMPUTER* OWNERS

*400/800/600XL/800XL

ATARI

- NEWS
- REVIEWS
- TUTORIALS
- UTILITIES
- HINTS & TIPS

plus more

THE BEST PROGRAM LISTINGS from

- U.S.A.
- U.K.
- AUSTRALIA

PUBLIC DOMAIN SOFTWARE LIBRARY

- SPECIAL OFFERS

PAGE 6 is published bi-monthly.
Annual Subscription is £7.00. Send TODAY to:

PAGE 6, P.O. BOX 54, STAFFORD, ST16 1DR

Tel. 0785 41153

NIGHTMARE REFLECTIONS

Runs in 16K Cassette or Disk

NIGHTMARE REFLECTIONS
NIGHTMARE REFLECTIONS

BY STEVE HILLEN

This is a sort of mini-adventure in which you must escape from a rather bad dream. In fact, there are about 344,064 different rooms inside so it's a pretty nasty nightmare !

For 400, 800 and XL machines

Once the program is running, you can reply to the prompt "Well ?" with the usual verb/noun structure eg: EXAMINE MIRROR

Some of these two word commands are jolly useful, so experiment around until some clues are revealed. The single letter commands understood are N,S,E,W for directions, Q for quit and I for inventory. All the clues are there so if you're logically minded, escape should be no more than 10 moves away.

Two hints for you are firstly read everything very carefully and secondly, try to think in terms of numbers. Oh, and don't bother with a map — since most dreams are hard to control, you will find that by going W then E, you may not end up where you started from.

Type in listings 1 and 2 if you are using a cassette, and listings 1 and 3 if you are using a disk. Save out both programs as one file before running it. The program will verify the data, notifying you of any typing errors. Then it will create either an autoboot cassette which can be loaded with or without Basic, or an Autorun Sys. file, depending on your system. The game is written in machine-code so that you'll have to work hard to extract any clues from the listings !

In the next issue, a subtle clue may appear in the Adventure Column, so if you're in trouble, keep your eyes peeled.

Good luck, and be logical. The odds are 1,048,576 to 1 against your stumbling out by chance !

```
2 REM **** Cassette version ****
65 IF PASS=2 THEN CLOSE #1:? "Done it.
":END
70 ? "Ready cassette and press <return
>";:OPEN #1,8,128,"C:":RESTORE 200:FOR
 X=1 TO 35:READ N:PUT #1,N:NEXT X
71 FOR A=36 TO 128:PUT #1,0:NEXT A
200 DATA 0,20,128,47,162,47,169,60,141
,2,211,169,70,141,231,2,133,14,169,57,
141,232,2,133,15,169,48,133,11
210 DATA 169,45,133,10,24,96
```

Listing 2

```
2 REM **** Disk version ****
65 IF PASS=2 THEN PUT #1,224:PUT #1,2:
PUT #1,225:PUT #1,2:PUT #1,45:PUT #1,4
8:CLOSE #1:? "Done it.":END
70 ? "Insert disk with DOS.Press <retu
rn>.";:DIM IN$(1):INPUT IN$:OPEN #1,8,
0,"D:AUTORUN.SYS"
90 PUT #1,255:PUT #1,255:PUT #1,0:PUT
#1,48:PUT #1,80:PUT #1,57
```

Listing 3

Listing 1

```
1 REM **** Nightmare Reflections ****
5 DIM Z$(1):Z$=CHR$(29)
10 DATA 0,1,2,3,4,5,6,7,8,9,0,0,0,0,0,
0,0,10,11,12,13,14,15
20 DIM DAT$(91),HEX(22):FOR X=0 TO 22:
READ D:HEX(X)=D:NEXT X:LINE=990:RESTOR
E 1000:TRAP 60:? "Checking data";Z$
25 LINE=LINE+10:? CHR$(28);"Line:";LIN
E:READ DAT$:IF LEN(DAT$)<>90 THEN 110
28 DATLINE=PEEK(183)+PEEK(184)*256:IF
DATLINE<>LINE THEN ? "Line :";LINE;" m
issing.":END
30 FOR X=1 TO 89 STEP 2:D1=ASC(DAT$(X,
X))-48:D2=ASC(DAT$(X+1,X+1))-48:BYTE=H
EX(D1)*16+HEX(D2)
35 IF PASS=2 THEN PUT #1,BYTE:NEXT X:R
EAD CHKSUM:GOTO 25
40 TOTAL=TOTAL+D1+D2+96:IF TOTAL>999 T
HEN TOTAL=TOTAL-1000
45 NEXT X:READ CHKSUM:IF TOTAL=CHKSUM
THEN 25
50 GOTO 110
60 IF PEEK(195)<>6 THEN 110
100 ? "Writing file":? :? :PASS=2:LINE
=990:RESTORE 1000:TRAP 60:GOTO 25
110 ? "Bad data on line:";LINE:LIST LI
NE:? TOTAL:END
1000 DATA 70707042002002700202020270002
0202F042000002020202020202020202024100
3048A9808D0AD48D18D06840,627
1010 DATA D8A238A06E20CB31A9308D0102A9
228D0002A9C08D0ED4A5581869E08D1230A559
69018D1330A9178554A90285,628
1020 DATA 558DC602A9008D3002A9308D3102
A214A9009D0006CA10FA8DC802A900AA9D0020
9D0021CAD0F7A92085F3A9F2,677
1030 DATA 85F2A206BD903585F1BD973585F0
BC9E3588B1F091F28B10F9A5F238E92885F2A5
F3E90085F3CA10DB200432A2,815
1040 DATA 00A94A9D4403A9369D4503A9069D
48038E4903A90B9D42032056E4A215A9209D80
06CA10FAA200A9059D4203A9,825
1050 DATA 069D4503A9809D4403A9149D4803
8E49032056E4C0001016AD0AD22901AABDA735
BCA535AA20CB3120E4314CB3,862
1060 DATA 30A200BD8006C99BF007E8E01490
F4B0DA8E0006E000F0D3E001D01CAD8006A205
DDDC35F005CA10F830C0BDE2,987
1070 DATA 3585F4BDE83585F56CF400A201BD
8006C920F008E8E01490F44CF930E88E0106E8
E8E88E0206A2008E04068E03,78
1080 DATA 06A000B98006DDEE35D008E8C8C0
0390F2B010EE0406AE0306E8E8E8E02890DE4C
F930A2008E05068E0306AC01,168
1090 DATA 06B98006C99BF005DD1536D009E8
C8CC020690EDB010EE0506AE0306E8E8E8E01C
90D84CF930AE0406BD3D3685,325
1100 DATA F4BD303685F56CF4008E45038C44
03A200A97F8D48038E4903A9098D42032056E4
60A2008E48038E4903A90B8D,349
1110 DATA 4203A99B2056E460AD0506C908F0
016068684CF930A202A9FF9D0606CA10FAA202
AD0AD22903A002D90606F0F4,400
1120 DATA 8810F89D0606CA10EBAD0AD22903
F0F9AACA8E0906A006A9B785F2A92085F3A900
8D0A0691F28810FB8C0606B9,552
1130 DATA 5036AC0A0691F2EE0A06EE0A06CA
10ECAD0AD22907F0F9A88C0B068B8ED036A000
BD5C36F007991A21E8C8D0F4,751
1140 DATA 991A21C8AD0AD22907AABD543618
6D0B063BE904C904B0F9BD0B06BDD736AABDB7
36F007991A21E8CBD0F4991A,934
1150 DATA 21C8C02590F8AD0AD22903AABD50
368DF920AD0AD2C940900CA025A90099422188
10FA3014A01FB9DF36994221,8
1160 DATA 8810F7AE0606BD50368D602160A2
36A0FF20CB314CCE33A237A01220CB314CCE33
20F731A200AD0506C903D006,40
1170 DATA AD0AD22901AABD2937BC2737AA20
CB314CCE3320F731AE2937AC273720CB314CCE
3320F731AD0506C902B006A2,160
1180 DATA 37A05F9004A237A07C20CB314CCE
3320F731AD0506C902B006A237A0BD9004A237
A0A220CB314CCE3320F731AD,204
1190 DATA 0506D007A237A0AF4C7133C901D0
07A237A0E24C7133A23BA01420CB314CCE33A2
38A02B20CB314CCE33A2008E,237
1200 DATA 4903A9388D4503A9588D4403A90B
8D4203A9118D4B032056E4E88E4B03CA8E4903
A9068D4503A9808D4403A905,248
1210 DATA BD42032056E4AD8006C959F00D20
E431A238A06A20CB314CCE334C2D3020E43120
D7334CB330AD0C06F8186901,269
1220 DATA 8D0C06AD0D0669008D0D06D8200
348D24208E2320AD0C062000348D26208E2520
60A829F04A4A4A4A186910AA,237
1230 DATA 98290F6910608AAE0906DD0606F0
0DCA10F8A238A07020CB314CCE33CD0B06D036
EE0E06AD0E06C90AD031A200,347
1240 DATA 8A9D50209D7820CAD0F7A238A0D4
20CB3120E431A239A02920CB31A9FF8DFC02AD
FC02C9FFF0F94C2D30A9008D,554
1250 DATA 0E06A238A06A20CB312004324CCE
3320F731AD0506C903D00AA238A08D20CB314C
CE33C904B00CAE2937AC2737,615
1260 DATA 20CB314CCE3338E904AC0906D906
06F0138810F8A238A0A38D0A0620CB31AD0A06
4C2734A238A08B20CB314CCE,706
1270 DATA 33AEE9E7E8F4EDE1F2E580B2E5E6
ECE5E3F4E9EFEEF300000000002D6F76657300
0310101010006279003300E00,775
1280 DATA 28696C6C656E0011191815290061
6D00696E006100736D616C6C0C007065726665
63746C790073717561726500,608
1290 DATA 726F6F6D2578697473006C656164
001A2D6972726F7273006C696E650074686500
6F746865720077616C6C730E,539
1300 DATA 00290063616E736565006D797365
6C66007265666C6563746564006D616E790074
696D65730E346F0074686500,395
1310 DATA 000073696465006F660074686500
666C6F6F72006973006100746965794343435
353535C0E5F71D294C6C2512,257
1320 DATA 260C232024A9C83535536F727279
2C2062757420492064 6F6E277420756E646572
7374616E642E9B45682D2079,161
1330 DATA 6F75277665206C6F7374206D6521
9B4E455357495111111111177B1343434343333
53484F53435257414B505553,973
1340 DATA 505245544F5550554C4558414C4F
4F52454144524F54414B464545574154504150
5348414D49524E2020452020,777
1350 DATA 532020572020202020323232343
4323333333333332DDDDE77474F1105353531F
39F157656C6C203F2E253337,588
1360 DATA 0109101924314051726564006F72
616E676500796 56C6C6F7700677265656E0062
6C756500696E6469676F0076,441
1370 DATA 696F6C6574006369726 36C650E00
747269616E676C650E007371756172650E0070
656E7461676F6E0E00686578,359
1380 DATA 61676F6E0E006865707461676F6E
0E006F637461676F6E0E006E6F6E61676F6E0E
0000040B12181D240008121A,326
1390 DATA 242D3740290063616E006665656C
0061006472617567687400667 26F6D00746865
00000E414141414172727272,77
1400 DATA 726767676868682021 9B4974277 3
206E6F74207468617420656173792021 9B2B3C
37374E6F7468696E67206861,971
1410 DATA 7070656E732E9B5468652 06D6972
726F7220737465616D7320757020746F207368
6F7720274E30272E9B426574,868
1420 DATA 746572206E6F742C2049206D6967
6874206E6565642069742E9B492063616E2774
20646F20746861742E9B4920,771
1430 DATA 616C726561647920686176652074
6861742E9B4974277320666978656420219B57
686174206120737472616E67,590
1440 DATA 652077617463682 02D2069742069
73206E756D6265726542066726F6D20302074
6F20332E9B54686520737175,420
1450 DATA 617265206269742 06F6620706170
6572207361797320 3A204D6F7665204F6E446F
776E20466F7572203F9B4920,295
1460 DATA 736565206E6F7468696E67207370
656369616C2E9B4920616D206361727279696E
67206120776174636820616E,177
1470 DATA 642061207069656 5206F662020
70617065722E9B4172652079 6F752073757265
20285929203F4F6B2E9B7D9B,53
1480 DATA 5468657265277320612 06D697272
6F7220696E2074686520776 1792E9B55736520
505553482022B206469726563,870
1490 DATA 74696F6E2E9B546865206D697272
6F7220666C69707320616E642E2E2E9B546865
72652773206E6F206D697272,857
1500 DATA 6F722074686572652E9B53756464
656E6C792C2061732069662049206861642070
75736865642074687 26F7567,713
1510 DATA 682020616E20696E76697369626C
6520666F7263652C492061776 16B6520696E20
6D792020202020262656472,534
1520 DATA 6F6F6D2E9B507265737320616E79
206B657920746F20706C617920616761696E2E
9B00000000000000000000,355
```

STARTING FROM BASICS

by Captain Hacker

Welcome to this new regular column. One of the most frequent requests we receive from our readers is for tutorials for those who are either just starting out with their computer, or who have problems understanding the rather spartan manuals supplied with it. Here we hope to help with a simplified, but thorough, explaination of several aspects of programming in BASIC in each issue. If you have any requests for help on particular commands or functions then please write to us we will be glad to here from you!

So here you are, you have just unpacked your computer, plugged it in, switched on, and in front of you is a blue screen with the word READY in the top left hand corner. Well what do you do? The manuals supplied with your Atari are hopelessly inadequate, but fear not for here we will try and start you on the road to enlightenment!.

PRINT

Firstly though, we must set ourselves a task. Lets suppose that we want the computer to simply print our name on the screen. Some how we have to tell the computer what we want, and we do this with something called a COMMAND. There are many different commands available to you, but the one which forces the computer to write on the screen is PRINT. Try typing the word PRINT on the keyboard, then press the RETURN key and see what happens. Not a lot, I hear you mutter! Well there is a very good reason for this, we have told it to print on its screen but we have not actually told it what we want it to print. Now type PRINT "JACK" and then press RETURN. This time you will see that the name JACK appears on your screen, but notice that the two quote (") characters either side of the name have not been printed. This is because we use them to tell the computer what we want printed, i.e. the computer only prints the characters between the two quotes.

Suppose we want the computer to keep repeating the name. We would have to give it the command each time – quite a laborious task! Fear not, for there is a better way, this is to change our command into a PROGRAM. Now type the following line, and press RETURN.

```
10 PRINT "JACK"
```

This time the computer does not appear to be obeying us, as it did previously. What is happening is that the computer first sees that we have given a reference number, and so it stores the line away in memory to be obeyed later, thus we have created a program!

Now type RUN and press RETURN. This is how we tell the computer to implement the commands it has stored in our program, and in this case it will print the name JACK. Now type in the following line, (remembering that line 10 is still in the computer).

```
20 GOTO 10
```

Press RETURN, type RUN, and see what happens. The computer looks at the lowest line number, (in this case 10) and executes the command on it, it then looks for the next line number, which is 20. Here it finds the GOTO command, which forces the computer to go back to the line number specified (in this instance it's 10).

We have put the computer into a loop from which it can never escape on its own! There are, however, two ways to stop the program – the first is to press the "BREAK" key. This stops the program in its tracks, and the computer will print a message telling you on which line BREAK stopped it. The other way is to push the SYSTEM RESET button. This is a far more drastic measure, however, since it does as the name implies and clears the screen and resets the system. You need not worry about what this does at the moment, except to remember that it is better to use the BREAK key whenever possible.

Type LIST and press RETURN, and you will again see your program. It should look like this: –

```
10 PRINT "JACK"
20 GOTO 10
```

Notice that when it was RUNning the name JACK was printed in a single column thus: –

```
JACK
JACK
JACK
JACK
etc.
```

Now re-type line 10, but before you press RETURN type a semi-colon after the closing quote, i.e.

```
10 PRINT "JACK";
```

Now type RUN again. This time the name is printed continually along each line, i.e.

```
JACKJACKJACKJACK....etc.
```

This demonstrates that the presence or absence of the semi-colon decides whether the next PRINT command will continue exactly where the last one finished, or on the next line instead. Try placing a comma in place of the semi-colon and see what happens. Experiment with different words between the quotes, or adding more PRINT commands. Remember that program line numbers can range from 1 to 32767, but it is common practice to use increments of 10 – this allows you to insert a line between two others easily. For example, in our little program we would insert the surname on the following line, i.e.

```
15 PRINT "JONES"
```

Remember that the computer will always place program lines in numerical order in its memory, regardless of the order you type them in.

One final point, there is a term we use to describe the collection of characters between the quotes. We call them STRINGS, a term which you will come across on many occasions, and I will describe this in more detail in the next issue.

PEEK and POKE

PEEK and POKE, despite being among the simplest of commands, are often the cause of much confusion to the beginner. He or she is often just told 'POKE this number here and this will happen', or 'the PEEK of 1785 plus 256 times the PEEK of 1786 will give this value....', so it is not surprising that these two commands appear magically powerful.

To understand PEEK and POKE well, you need to have a reasonable grasp of

what your computer's memory actually is. A good analogy is to imagine a long row of buckets, each with its own reference number. Since there are 65536 possible memory locations these would be labeled from 0 to 65535. Each of these imaginary buckets is able to store a single number, but with the limitation that this number can only have a value of 0 to 255. This may at first seem quite restrictive since we ourselves might want to store numbers of a much higher value, several thousands perhaps. The way around this is to use several of these buckets together to represent a much larger value, and this is what BASIC does for you. Atari BASIC actually uses six buckets to store each number you give it, and it converts your number into a specially coded form, first to allow you to have both extremely large numbers as well as very tiny fractions!

However when you use these buckets with the Peek and Poke instructions to alter the way the machine works, you will usually deal with them as single bucket values or two buckets. When two buckets are used in combination, what happens is that we use the first as our single units of count, and each time we reach 256 we add one to the bucket with the highest numbered label of the two, and clear the other. Thus the overall value of the two combined is usualy taken as:–

THE SECOND BUCKET CONTENTS MULTIPLIED BY 256, PLUS THE FIRST BUCKET'S CONTENTS.

So how can we look at, or even alter the contents of these memory locations, or buckets as we are calling them? To look at the contents of a bucket we must use the command PEEK. For example:–

```
PRINT PEEK(84)
```

This will print the contents of bucket number 84, which happens to be the one which the computer uses to keep note of the cursors current vertical position. Conversely, we can use the POKE command to change the value held in location 84. For example:–

```
POKE 84,10
```

Whilst using the PEEK command you need not worry about crashing your system - PEEK only looks, it does not alter. POKE, however, is a different matter, although it is not possible for this command to permanently damage your computer, it can, if used incorrectly, cause your computer to lock-up. Not a happy event if you were in the middle of writing a large program!

The value that each of these memory locations holds is called a byte. Most of these locations are alterable, and are refered to as Random Access Memory or RAM for short. There are some locations which are not alterable since their contents are fixed permanently at the factory. These are called Read Only Memory, or ROM, and they usually hold the programs and data required to run the computer since this must not be lost when the machine is switched off.

The third type of memory location is called a hardware register, some of these can only be written into and others can only be read from. Hardware registers are the ones which are actually 'connected' to the parts they control – and this is usually some form of electronic switch inside your computer.

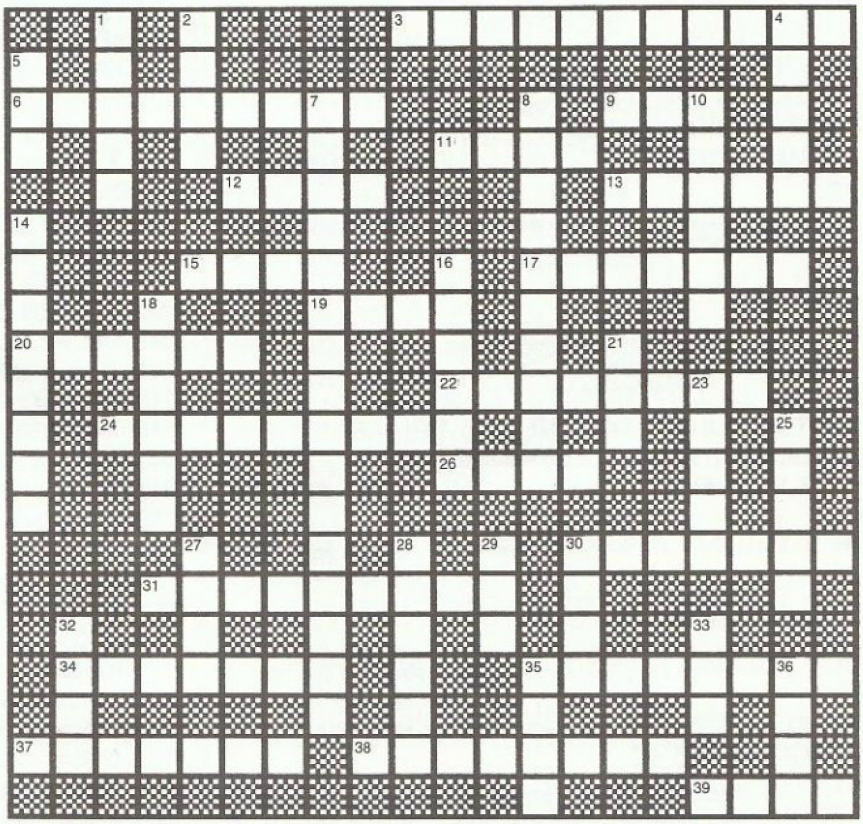
Used wisely, PEEK and POKE can add a great many interesting and useful functions to your programs, and I would recommend that you purchase a book or listing of some kind giving all the locations and their purposes – there are several available.

CROSSWORD COMPETITION

Complete the puzzle and send your answers to us, as soon as possible. The closing date that answers must be received by is the 30th April, as on the 1st of May the first person drawn with all the correct answers wins **TEN POUNDS.** It may be that none of you will be able to complete our crossword, therefore please send your answers in, even if not completely finished. In this case we will give the prize to the person with the most correct answers. It will also tell us if the clues in our crossword are too hard.

As we don't think you will want to ruin the look of your magazine, we suggest that you send your answers, in numerical order on a separate piece of paper.

Answers to: **'CROSSWORD'**
The UK Atari Computer Owners Club, P.O. Box 3, Rayleigh, Essex.



ACROSS CLUES

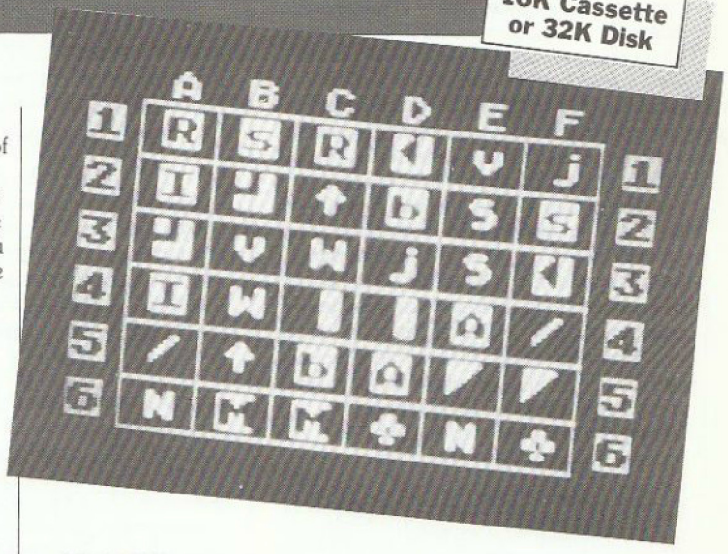
3. El Supriemo (4,7)
6. An essential link (9)
9. It never forgets (3)
11. Watch out for teeth marks! (4)
12. Shows your program (4)
13. Computer language (6)
15. Memorable boundary (4)
17. To be obeyed (7)
19. A small 'C'! (4)
20. A floppy block (6)
22. No hope? (8)
24. Rotates your flexible record (4,5)
26. We start-up in 0 of this (4)
30. We count on this! (7)
31. Could swear on these routines (9)
34. How 7 down operates (3,4)
35. Mind your basic (8)
37. Well rounded figure! (7)
38. Takes quite a bashing (8)
39. Initiate a file operation (4)

DOWN CLUES

1. I.C. a magazine! (5)
2. Another round figure (4)
4. We could get trapped into this (5)
5. A little piece (3)
7. A slothful peripheral (8,8)
8. Over-used term for a good program (10)
10. Handy documentation (6)
14. Maybe a game addicts syringe? (8)
16. Complete computer layout (6)
18. Strange characters que for it! (6)
21. Not a very bright command! (3)
23. You must all have one! (5)
25. Numbers with a common link (5)
27. For this we loop (4)
28. Name for 4 bits (6)
29. Begin again with this (3)
30. A much-processed material! (4)
32. Not odd at-all (4)
33. It crawls into programs (3)
35. Lets go round again! (4)
36. Your first program, may be? (4)

MATCHBOX

Runs in 16K Cassette or 32K Disk



by Peter Cunningham — Chester.

Improve your concentration by having a go at this game, find the hidden pairs of shapes to win. You will be shown the positions of all the pairs at the beginning of the game and then they are erased and it is up to you to remember where they were. A six by six grid is drawn and the shapes are positioned randomly in the grid. After the shapes have been erased, one shape will be displayed and then you are given a joystick controlled cursor to place over the square where you think the match is. When you press the button your choice is examined and if you have guessed correctly the two shapes are left showing and points are awarded. If you guess incorrectly the computer has a little chuckle to itself and moves on to the next selection. *For 400, 800 and XL machines*

In this program, anything which is underlined, should be entered in 'INVERSE'.

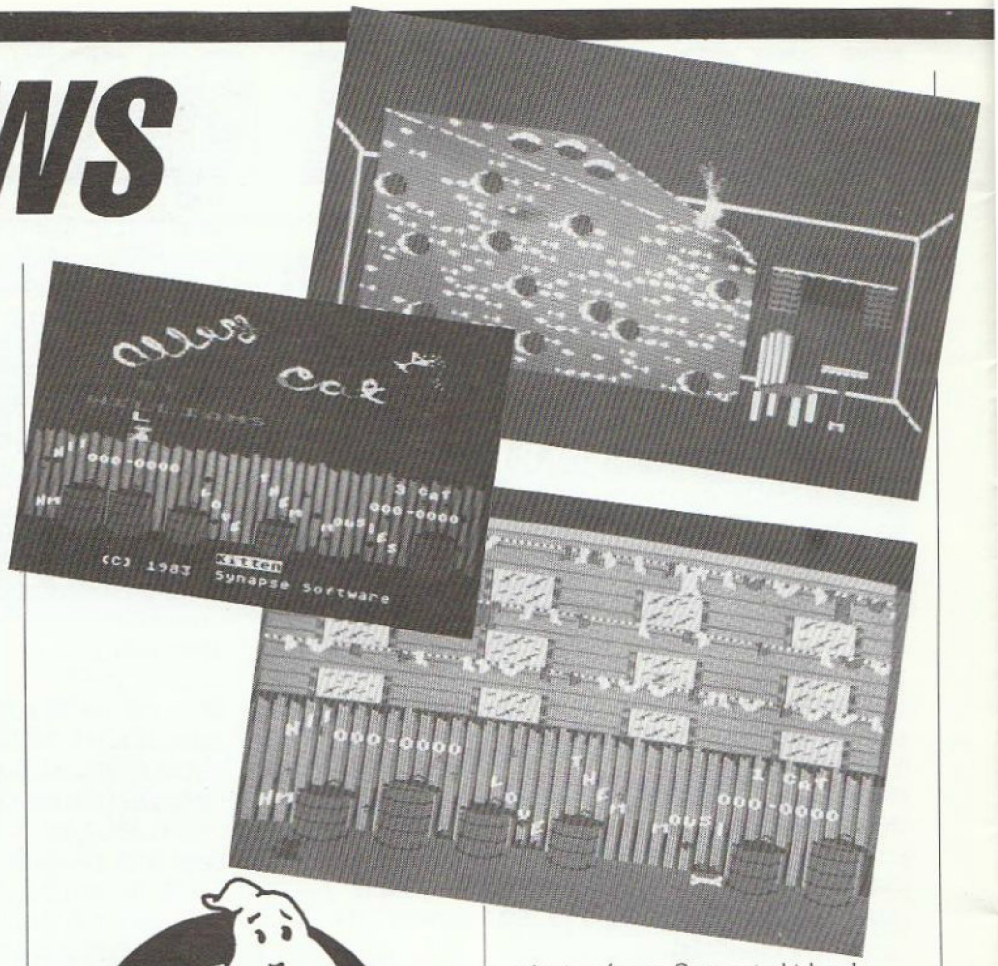
```
50 J=0:ML=1536:IF PEEK(ML)=J THEN RESTORE 20010:GOSUB 20
000
100 GOSUB 9000:POKE 559,S:CLR :POKE 712,96:J=0:DIM Z$(1)
:Z$=CHR$(125):GOTO 6000
410 SEL1=INT(RND(J)*6+1):SEL2=INT(RND(J)*6+1)
415 IF CHOSEN$(B(SEL1,SEL2),B(SEL1,SEL2))="Z" THEN 410
417 CHOSEN$(B(SEL1,SEL2),B(SEL1,SEL2))=T$(B(SEL1,SEL2),B
(SEL1,SEL2))
420 REM  DRAW CARD TO FIND IN C,2
421 POKE 77,J:CL=2:GOTO 3000
434 SOUND J,REF(SEL1,SEL2)*3+50,10,10:CL=1:REM  CHOSEN C
ARD
435 A=USR(1536,ASC(T$(B(SEL1,SEL2),B(SEL1,SEL2))),CL,2+2
*SEL2,10+12*INT(REF(SEL1,SEL2)/6))
441 POKE 53278,1:? Z$;"                              ";S$;HIT;"  "
;B$;MISS
442 IF FL=J THEN FLA=ADR(M2$):CH=1
443 IF FL=5 THEN FLA=ADR(M3$):CH=-1
444 SOUND J,J,J,J:IF STRIG(J)=J THEN 500
447 ST=STICK(J):POKE 53278,1:IF ST=15 THEN A=USR(MISL,2,
PMB,FLA,XSTIK,YSTIK,7):FL=FL+1*CH:GOTO 442
448 IF ST=7 THEN XSTIK=XSTIK+3:IF XSTIK>163 THEN XSTIK=7
6
449 IF ST=11 THEN XSTIK=XSTIK-3:IF XSTIK<76 THEN XSTIK=1
63
450 IF ST=14 THEN YSTIK=YSTIK-3:IF YSTIK<25 THEN YSTIK=8
9
451 IF ST=13 THEN YSTIK=YSTIK+3:IF YSTIK>89 THEN YSTIK=2
5
455 SOUND J,XSTIK+YSTIK,10,8:A=USR(MISL,2,PMB,ADR(M1$),X
STIK,YSTIK,7):GOTO 442
460 ? Z$;" you have FOUND    this shape already!":FOR D=
1 TO 150:SOUND J,XSTIK*YSTIK/D,10,10:NEXT D:GOTO 441
461 ? Z$;"                          CORRECT!":HIT=HIT+1:FO
R S=5 TO 25:A=USR(ADR(R$)):SOUND J,XSTIK*YSTIK/S,10,10
462 FOR X=J TO 35:NEXT X:NEXT S:POKE 705,230:POKE 706,13
6:POKE 707,232:POKE 708,40:POKE 709,J:POKE 710,182
463 POKE 711,70:POKE 704,198:CHOSEN$(B(SEL1,SEL2),B(SEL1
,SEL2))="Z"
464 CL=3:GOTO 3000
467 ? Z$;"                          missed!":MISS=MISS+1:F
OR SO=J TO 250 STEP 10:SOUND J,SO,12,10:NEXT SO
469 POKE 709,220:FOR T=28 TO 1 STEP -0.24:SOUND J,YSTIK*
XSTIK/T,14,8:NEXT T:POKE 709,J
475 CL=J:GOTO 3000
500 A=USR(MISL,2,PMB,ADR(M0$),XSTIK,YSTIK,7):ON PEEK(532
50) GOTO 447,461,447,460,447,447,447:GOTO 467
2022 IF CL=3 AND B(SEL1,SEL2)=B(ROW,COL) THEN SOUND J,RE
F(SEL1,SEL2)*3+200,10,10
3000 A=USR(MISL,2,PMB,ADR(M1$),XSTIK,YSTIK,7):SOUND J,J,
J,J:FOR ROW=1 TO 6:FOR COL=1 TO 6
3020 IF B(SEL1,SEL2)=B(ROW,COL) THEN A=USR(1536,ASC(T$(B
(SEL1,SEL2),B(SEL1,SEL2))),CL,2+2*COL,10+(ROW-1)*12)
3022 IF CL=3 AND B(SEL1,SEL2)=B(ROW,COL) THEN SOUND J,RE
F(SEL1,SEL2)*3+140,10,10
3030 NEXT COL:NEXT ROW:IF HIT>17 THEN 5000
3032 IF CL=3 OR CL=J THEN 410
3040 GOTO 434
5000 R=(HIT+MISS)/HIT:IF R<1.5 THEN MM$="ACE"
5010 IF R>=1.5 THEN MM$="WOW!"
5020 IF R>=2 THEN MM$="HOT"
5022 IF R>=2.5 THEN MM$="GOOD"
5024 IF R>=3 THEN MM$="AVE."
```

MATCHBOX

```
5026 IF R>=3.5 THEN MM$="FAIR"
5028 IF R>=4 THEN MM$="POOR"
5030 IF R>=4.5 THEN MM$="BAD!"
5032 IF R>=5 THEN MM$="YUK!"
5040 ? Z$;"you missed: ";MISS;"       MEMORY IS: ";MM$
5050 POKE 709,70:FOR I=1 TO LEN(EDD$):SOUND J,I*3,10,12:
A=USR(1536,ASC(EDD$(I,I)),2,I-1,0):FOR D=J TO 15:NEXT D
5055 NEXT I:A=USR(MISL,2,PMB,ADR(M0$),250,J,6):SOUND J,J
,J,J
5056 A=USR(ADR(R$)):FOR L=J TO 150:NEXT L:IF PEEK(53279)
<>6 THEN 5056
5070 POKE 77,J:RUN
6000 DIM B(7,7),I(18),T$(18),X$(6),Y$(6),S$(6),B$(8),REF
(7,7),CHOSEN$(18),XG$(1),MM$(4),ST$(1),EDD$(20),R$(27)
6010 RESTORE 30100:FOR L=1 TO 27:READ B:R$(L,L)=CHR$(B):
NEXT L
6030 RESTORE 30000:DIM MISMOV$(114):MISL=ADR(MISMOV$):FO
R X=1 TO 114:READ N:MISMOV$(X)=CHR$(N):NEXT X
6050 DIM M0$(7),M1$(7),M2$(7),M3$(7):FOR I=1 TO 7:READ N
:M0$(I)=CHR$(N):NEXT I
6060 FOR I=1 TO 7:READ N:M1$(I)=CHR$(N):NEXT I
6070 FOR I=1 TO 7:READ N:M2$(I)=CHR$(N):NEXT I:FOR I=1 T
O 7:READ N:M3$(I)=CHR$(N):NEXT I
6090 PMBASE=INT((PEEK(145)+3)/4)*4:POKE 54279,PMBASE:PMB
=PMBASE*256:EDD$="Press START to play!"
6110 POKE 559,46:POKE 53277,3:POKE 53260,255:POKE 623,1:
POKE 704,198:POKE 705,230:POKE 706,136:POKE 707,232
6180 XSTIK=65:YSTIK=16:FOR L=J TO 18:I(L)=J:NEXT L:T$="
                ":T=1
6200 Z=J:R=PEEK(53770):IF R=14 OR R=39 OR R=46 OR R=96 O
R R=108 OR R=95 OR R=44 THEN 6200
6210 IF R=9 OR R=11 OR R=12 OR R=15 OR R=2 OR R=22 OR R=
32 OR R=90 OR R=13 THEN 6200
6220 ST$=CHR$(R):FOR X=1 TO 18:IF T$(X,X)=ST$ THEN Z=1
6250 NEXT X
6260 IF Z=1 THEN 6200
6270 T$(T,T)=ST$:T=T+1:IF T>18 THEN 6290
6280 GOTO 6200
6290 CHOSEN$=T$
6300 FOR N=1 TO 6:FOR M=1 TO 6
6320 S=INT(RND(J)*18+1)
6330 IF I(S)=2 THEN 6320
6340 I(S)=I(S)+1:B(N,M)=S:NEXT M:NEXT N
6370 GRAPHICS 7:POKE 16,64:POKE 53774,64:POKE 710,J:POKE
 752,1:DL=PEEK(560)+PEEK(561)*256
6390 IF PEEK(DL)<>66 THEN DL=DL+1:GOTO 6390
6400 POKE DL,71:POKE DL+3,7:POKE DL+4,65:POKE DL+5,PEEK(
DL+7):POKE DL+6,PEEK(DL+8)
6410 COLOR 1:FOR X=8 TO 70 STEP 12:PLOT 27,X:DRAWTO 123,
X:NEXT X:PLOT 27,79:DRAWTO 123,79
6420 COLOR 1:FOR X=8 TO 70 STEP 12:PLOT 27,X:DRAWTO 123,
X:NEXT X:PLOT 27,79:DRAWTO 123,79
6430 FOR Y=28 TO 125 STEP 16:PLOT Y-1,80:DRAWTO Y-1,8:NE
XT Y
6440 X$="ABCDEF":FOR PR=1 TO LEN(X$):A=USR(1536,ASC(X$(P
R,PR)),1,2+PR*2,J):NEXT PR
6450 S$="HIT: ":B$="miss! ":Y$="123456":FOR SIDE=2 TO 16
 STEP 14
6460 FOR PR=1 TO LEN(Y$):A=USR(1536,ASC(Y$(PR,PR)),1,SID
E,PR*12-2):NEXT PR:NEXT SIDE
6470 POKE 559,46:POKE 53277,3:CL=2
6480 FOR N=1 TO 6:FOR M=1 TO 6
6500 A=USR(1536,ASC(T$(B(N,M),B(N,M))),CL,2+M*2,10+(N-1)
*12):IF CL=2 THEN REF(N,M)=TAL:TAL=TAL+1
6520 NEXT M:NEXT N:IF CL=2 THEN FOR X=500 TO 2 STEP -1:S
OUND J,X/2,12,8:FOR Z=J TO 7:NEXT Z:NEXT X:POKE 710,182
6530 IF CL=2 THEN CL=J:GOTO 6480
6540 POKE 709,J:GOTO 410
9000 GRAPHICS 23:S=PEEK(559):POKE 559,J:DIM A$(220),F$(5
5):RESTORE 10000:FOR A=1 TO 4:READ F$:A$(A*55-54,A*55)=F
$:NEXT A
9010 L=1:M=J:FOR Y=J TO 87 STEP 8:FOR I=J TO 19:A=USR(15
36,ASC(A$(I+L,I+L)),2,I,Y):NEXT I
9020 M=M+1:L=M*20+1:NEXT Y:RETURN
10000 DATA ..........................................
..........
10001 DATA ......Welcome.....................To.........
.....MATCH
10002 DATA -FOX.....................................
..........
10003 DATA ................Get ready.................
..........
20000 FOR I=J TO 252:READ A:POKE ML+I,A:NEXT I:RETURN
20010 DATA 104,240,10,201,4,240,11,170,104,104,202,208,2
51,169,253,76,164,246
20040 DATA 104,138,195,104,201,128,144,4,41,127,198,195,
170,141,250,6,224,96,176,15,169,64,224,32
20080 DATA 144,2,169,224,24,109,250,6,141,250,6,104,104,
141,251,6,104,104,141,252,6,14,252,6
20120 DATA 104,104,141,253,6,133,186,166,87,169,10,224,3
,240,8,169,20,224,5,240,2,169,40,133
20160 DATA 207,133,187,165,88,133,203,165,89,133,204,32,
228,6,24,173,252,6,101,203,133,203,144,2
20200 DATA 230,204,24,165,203,101,212,133,203,165,204,10
1,213,133,204,173,250,6,133,187,169,8,133,186
20240 DATA 32,228,6,165,212,133,205,173,244,2,101,213,13
3,206,160,0,162,8,169,0,133,208,133,209
20280 DATA 177,205,69,195,72,104,10,72,144,8,24,173,251,
6,5,208,133,208,224,1,240,8,6,208
20320 DATA 38,209,6,208,38,209,202,208,228,104,152,72,16
0,0,165,209,145,203,200,165,208,145,203,104
20360 DATA 168,24,165,203,101,207
20370 DATA 133,203,144,2,230,204
20380 DATA 200,192,8,208,183,96
20390 DATA 169,0,133,212,162,8
20400 DATA 70,186,144,3,24,101
20410 DATA 187,106,102,212,202,208
20420 DATA 243,133,213,96,0,1,28
30000 DATA 216,104,104,104,133,213,104,133,206,104,24,10
5,128,133,205,165,206,105,1,133,206,104,133,204,104
30010 DATA 133,203,104,104,133,208,104,104,133,209,104,1
04,24,101,209,133,207,160,0,162,0,134,212,169,252
30020 DATA 166,213,240,7,10,10,9,3,202,208,249,166,212,4
9,205,145,205,196,209,144,30,196,207,176,26
30030 DATA 132,212,138,168,177,203,164,213,240,5,10,10,1
36,208,251,164,212,17,205,145,205,232,169,0,240
30040 DATA 0,200,192,128,208,196,166,213,165,208,157,4,2
08,96
30050 DATA 3,3,3,3,3,3,3,3
30060 DATA 3,0,0,0,0,0,0,3
30070 DATA 3,1,1,1,1,1,1,3
30080 DATA 3,2,2,2,2,2,2,3
30100 DATA 104,162,0,172,192,2,189,193,2,157,192,2,232,2
24,8,144,245,140,199,2,96,65,65,65,65,65,65
```

REVIEWS

by Ralph Kingsley



ALLEY CAT

The idea of having a game all about cats seems pretty stupid on the face of it, but Alley Cat from Synapse Software is so well written and so exciting to play that my doubts soon vanished after only a few minutes. Alley Cat is fast action entertainment from start to finish. You are Freddy the Cat and are busy getting into all sorts of trouble, whilst trying to avoid Bowser von Spike the big bully of a dog that 'hounds' you at every turn. You start each adventure outside the fence, in the alley which is strewn with dustbins and litter, your object is to jump onto the bins and from there up onto the top of the fence, this action avoids Bowser who patrols the alley and also gives you the opportunity to leap onto the washing lines. The washing lines are infested with mice, and by leaping from line to line you can score points for every mouse you catch, but beware do not stay still for to long because a mouse will dash out and unpeg the garment on which you are hanging and you will crash to the ground. The washing lines also move in the breeze and gradually the washing is dislodged so making leaping from line to line much more difficult.

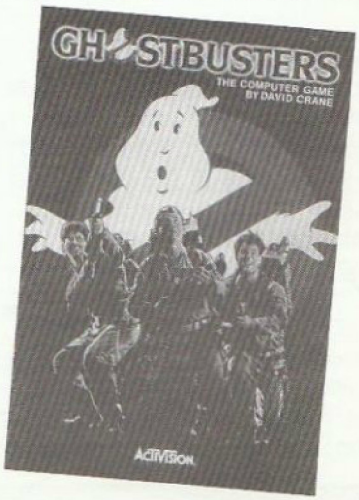
Behind the washing lines are many windows which open at random times, if you can manage to drop into one of these open windows you move into another scene of action. It is not always easy to drop in though as quite often objects are thrown at you, such as bones, bottles, shoes and telephones. Once inside various new adventures await you. Most rooms contain a maniac broom which chases you and gives you a clout if it can, to slow it down you can lay down dirt on the floor which it must sweep before coming after you. Each room has a goal for you to achieve such as diving into a goldfish bowl and gobbling up as many as you can, or catching mice in a giant cheese, or drinking the milk from bowls in a room full of sleepy dogs, or climbing to the top of a bookcase to knock down the flowers. If you gain your objective you are allowed to try and reach the lady cat to receive your reward!

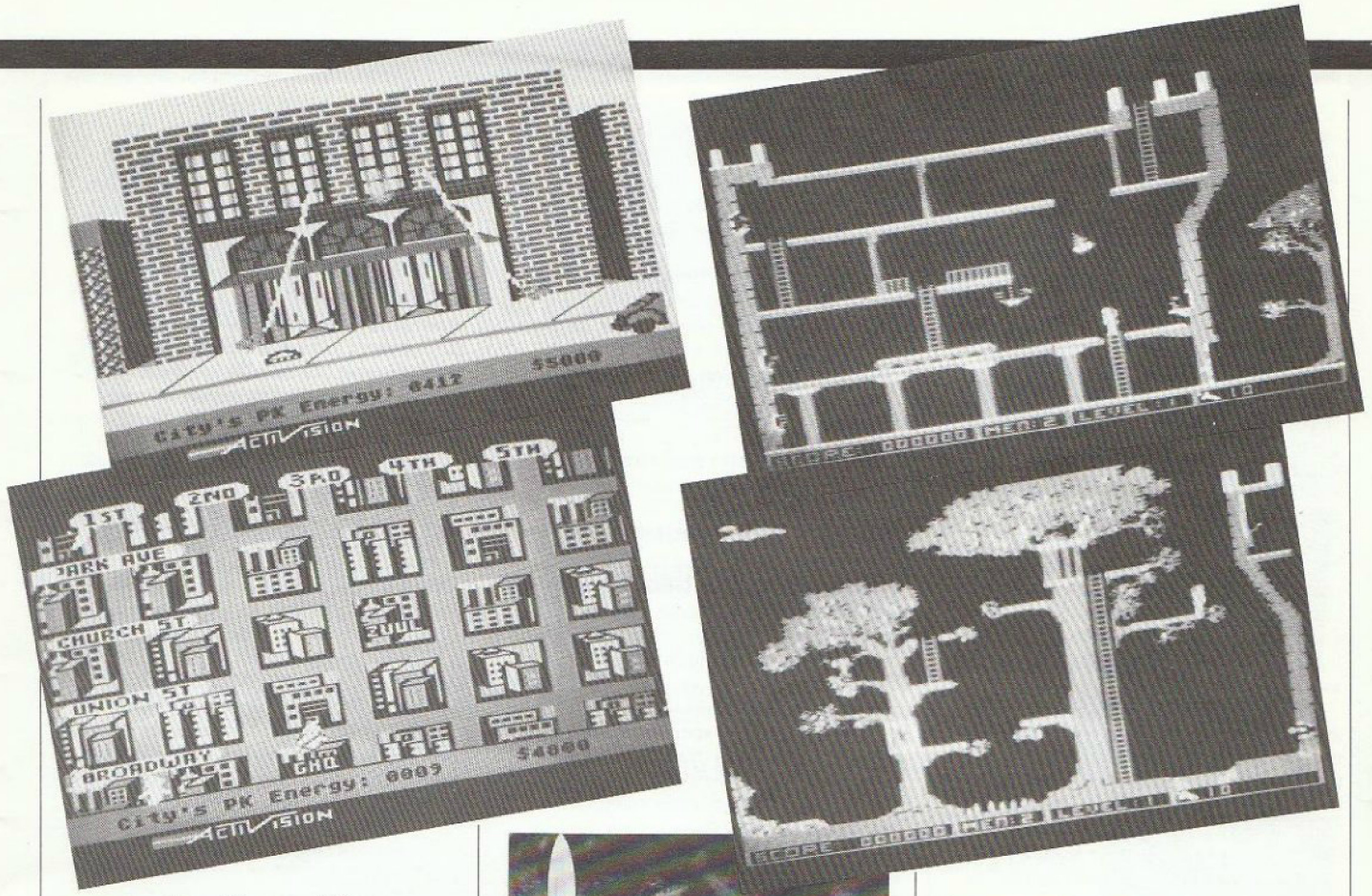
There are three lives available to Freddy and you can select your level of play, you can start at Kitten level, move up to Housecat, then on to Tom Cat, and finally upto Alley Cat, each level getting progressively more difficult. Control of Freddy is by joystick and movement is fast and furious. Sound effects enhance the game, my favourite noises being the growling of the dogs and the screech of the cat when fighting with Bowser. All in all an excellent game, it has replaced Boulder Dash as my favourite. My latest information is that Alley Cat will be available in the U.K. soon under the Synsoft label, but formats and prices are unknown.

GHOSTBUSTERS

It is getting to be tradition to make computer game versions of popular films and Ghostbusters is no exception. The scenario is that many ghosts are trying to congregate at the Temple of Zuul in order to create a disaster of biblical proportions, and only you can save the city. But before you can fight the ghosts you have to kit your ghostbusting team with all the paraphernalia required to catch and cage ghosts. To do this you are given a loan by the bank in order to purchase the equipment, which includes a PK Energy Detector which warns of an approaching ghost or 'Slimer' by turning a building pink as you pass it. You can also buy a Marshmallow Sensor, an Image Intensifier which makes Slimers easier to see, a Ghost Vacuum which sucks up 'Roamer' ghosts as you travel about the city, or a Portable Laser Confinement System which stores ten Slimers so eliminating the need to return constantly to GBHQ. The most important items you can buy are the Ghost Traps (without which you cannot capture ghosts) and your Ghostbusting Vehicle, there is a choice of four cars with varying price tags from a Compact which only carries 5 items, a Hearse, a Station Wagon or the top of the range High Performance car which can travel upto 160 MPH. Note however that whatever you buy you must end the game with more money than you started with!

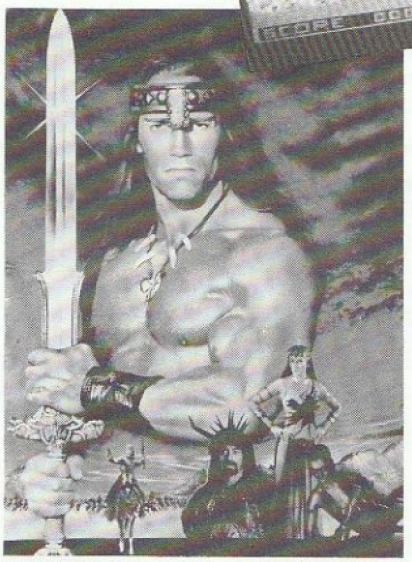
Now that you are laden with equipment you can venture out into the city streets in search of Slimers, Roamers and dreaded Marshmallow Men. The city blocks are divided into buildings which turn red if occupied by a ghost. You are represented by a little Ghostbuster symbol and as you move along the streets to the red buildings you leave a dotted trail. In addition Roamers are making their way to the Temple of Zuul, if you move your pointer over one it will freeze, then when you start to drive through the streets you can suck it up with your Vacuum, your PK





energy counter will increase. When you arrive at a haunted building two of your men (you have 3) will disembark and set a trap for the ghost, one man should be sent to one corner of the building and the other man is positioned on the opposite side, by pressing the fire button their Negative Ioniser Backpacks will be powered up. You have to move your men in from both sides gradually drawing the Slimer into the trap, when you feel the time is right press the button again and the Slimer will be pulled down into the trap, be very careful though because the ghost could 'slime' one of your men and escape. Every ghost you catch is rewarded financially and in this way you can improve and replenish your equipment.

If you survive the game this far, you get a chance to sneak at least two of your men into the Temple of Zuul which is being guarded by the Marshmallow Man by running between his legs and entering the building. Thus by reaching the top of the building you have successfully ended the game. Overall this new game from Activision is very enjoyable, but there are one or two niggles! For instance, the Commodore version employs voice synthesis to enhance the game, but not so on the Atari version (maybe the guy who converted it did'nt know you could do it on the Atari?). Also it seems to be too slow at the start and too fast at the end. In the beginning the hauntings are few and far between and at the close upto 4 houses at once are red and 5 more are pink, enough to scare the hell out of you! But these are only minor points really, in reality I am sure Activision are on a winner with this one. Ghostbusters will be available on disk in March for £14.99.



CONAN

The trend of making a film of a book has been carried one step further by Datasoft, they have produced a computer game all about the adventures of Conan the Cimmerian. This particular adventure is sub-titled "Hall of Volta" in which our hero must find his way to Volta's Lair, avoiding all the hazards of the journey, to destroy the evil mage who dwells there. Conan is helped in his task by a friendly bird who guides him through the maze of passageways. Conan must pass water barriers, hot geysers, poisonous scorpions, dragons and other monsters, and on the way he needs to find keys and gems which will help him to get to his goal. Conan has 3 lives to play with and ten swords, although more swords can be found, the swords are used to kill attacking creepy crawlies and gain points. The swords are thrown and return rather like a boomerang. If you miss however you loose one of your swords. One of the novel features of this game is the way Conan somersaults through the air landing safely on his feet.

The graphics are excellent and very reminiscent of the Bruce Lee graphics, it could be because they are both written by Ron J. Fortier, so if you have seen Bruce Lee you'll know what to expect. Conan is being released in the U.K. on the US Gold label sometime in March on cassette for £9.95 and for £14.95 you can get the disk version. When it is out I am sure it will be a big hit, it certainly deserves to be.

SPY Vs SPY

If you have ever seen the American comic MAD, you will know very well of the zany antics of SPY vs SPY, two secret agents one dressed in white, the other all in black, who constantly do battle with each other. First Star Software have faithfully reproduced the cartoon characters into this computer game and are to be congratulated on also capturing the madcap humour of the Spy's. The game can be played against the computer or another opponent, and your mission is to escape from the opposition's embassy carrying the Top Secret Briefcase containing a Passport, travelling Money, Secret Plans, and a Key, and you must do this before your time runs out because your aircraft leaves at a set time with or without you. The screen display shows two rooms in the embassy, the one where the White Spy is, and the other where the Black Spy is

operating, and when they are in the same room the other room blanks out temporarily. Next to each room display is the "Trapulator" which shows all the booby-traps available to each Spy, you can also use it to pinpoint your location, check inventory and gauge your time. Thus not only do you have to search for the hidden secrets but you can lay traps to snare your opponent. There are 5 traps you can use, these are a Bomb, a Spring, a Water Bucket, a Time Bomb and a Gun with a String. For most of the traps there is a remedy, such as a Water Bucket will defuse a Bomb, or a pair of Scissors will disable the Gun with the String. You can call up a map of the embassy which will help you to locate the secrets you are looking for, but you will loose points for using it. The Trapulator will display the items you have successfully collected. Once you have got all you need you must leave by the only exit and make your way to the airport to make your escape.

There are several levels of play and on the higher levels the embassy can have more than one storey. To get to the upper floors you can raise and lower a ladder through a hole in the floor or ceiling, note that in many instances the stairways are hidden under a rug. Another feature is that you can leave a trail of "Bread Crumbs" so that you can retrace your steps to your starting point. When both spies are in the same room they can fight each other by weilding a club and trying to hit their opponent on the head or body. About 7 good solid blows will "Kill" him, however he will return in time so don't think you have won.

Spy vs Spy is an excellent game which holds the interest and is fun to play, especially in the two player mode. Beyond Software publish Spy vs Spy, under licence from First Star, for the Spectrum and Commodore 64. As far as is known Beyond have no plans to publish the Atari version. *Pity.!*

READERS LISTING SERVICE

Introducing An Exciting New Service For Our Readers.

It is certain that all budding programmers have experienced the sheer frustration of trying to trace through a program by listing parts of it to the screen. Lets face it, it's a nightmare trying to remember values assigned in one section after you have jumped off to look at a subroutine at the other end of your program! Of course, for those of you who own a printer this is no problem – you just dump a copy of the program to your printer, pour yourself a coffee, sink into your most comfortable armchair and relax while you study your listing. Or perhaps take it to work to study in your lunch break, and perhaps ask a freinds opinion. The trouble is, though, that printers are definitely not cheap, since a reasonable quality dot matrix printer would cost around three to four hundred pounds. Add to this the cost of a suitable interface and you have a considerably expensive piece of hardware, beyond the scope of most enthusiasts!

Fear not, dear friends, for help is at hand. Just send us your program on cassette or disk together with £2 and we will return your cassette or disk with a PRINTED listing of your program!

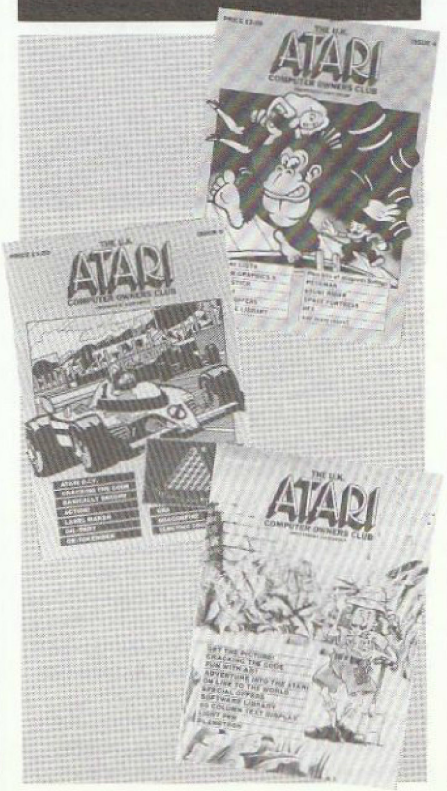
Please state whether you would like your program listed in 38, 40 or 80 column width, and remember that control characters obtained using the CONTROL key cannot be printed (We will just replace them with a dummy character).

Also note that the club reserves the right to refuse to supply this service if it is considered that the length of material is excessive.

Send your cassette/disk to:
'LISTING SERVICE'
The U.K. Atari Computer Owners Club,
P.O. Box 3, Rayleigh, Essex.

BACK ISSUES

Previous issues of this magazine are obtainable from the club for £1 plus 30p. postage each. They contain many interesting and informative articles, hints & tips, program listings for you to input, reviews and practical advice. If you have missed out send for your copies of back issues today! Please note that issues 1, 2, and 3 are already sold out.



Issue 4.

Includes a complete in-depth look at Display Lists, what they are, how to use them, LMS explained, horizontal and vertical scrolling, etc. Another article shows how to get text on a Graphics 8 screen and gives an example graph to prove the point. A comprehensive review of many of the different types of joystick that are available gives ratings for comfort, action, looks and value. Program listings are aplenty and include Peckman, a Basic version of a well-known arcade game, Stunt Rider, in which you must jump your motorbike over the buses, Hex is a two player board game with excellent graphics, and for the more serious minded, you can enjoy designing your own shapes with CAD (computer assisted design).

Issue 5.

The first part of the series on 'Cracking the Code' starts in this issue and covers Binary, Hexadecimal and Decimal mathematics. There is an article on protecting your Basic programs from prying eyes and an interesting article on hardware modifications to the 800/400 machines to give improved sound and picture quality, a cold start key and a busy light for your cassette player. Also included is a review of the new programming language. 'Action!' showing its potential for creating exciting fast action games. Games listings shown include Gil-bert, which is a 'Q-bert' type game, also Dragonfire in which the player must cross the drawbridge dodging the dragons flaming breath to reach the treasure room. Other listings include a label maker and a QRA locator for Radio Amateurs.

Issue 6.

Includes a useful tutorial showing how to print Micropainter and Versawriter pictures, also contains a terrific program demonstrating 80 characters across the screen. A new regular column for adventure enthusiasts is started to give reviews of adventure games and give hints and tips on how to play them. Part two of Cracking the Code continues with addressing modes and binary sums. The hardware design for a Light Pen is shown together with some simple programs to use with it once you have built it. Fun with Art from Epyx is reviewed and some of the excellent results of using this package are shown. Programs include Planetron and an RTTY listing for use with a short wave band radio, the Atari 850 interface and a signal terminal unit (such as the Maplin TU1000).

31 STONEYHURST ROAD, ERDINGTON, BIRMINGHAM
TELEPHONE:- (021) 384 5080

Dear Atari User,

So you think 'SOFTWARE EXPRESS" is just another mail order company, do you? Well you are WRONGGG!!!

Our Atari A—Team of computer industry experts will provide you, the user, with the most comprehensive service available in the U.K. World-wide contacts, products knowledge and experience ensure that we can obtain any Atari products. Our boast is quite simple:

"IF IT'S AVAILABLE, ANYWHERE IN THE WORLD, WE WILL GET IT"

And not just software - hardware, printers, cables, modems, magnetic media — in fact, you name it and we can get it! Or why not take out subscriptions to specialist magazines? We can also supply individual copies on request.

And, no, we haven't forgotten books! Choose from a wide range or we'll be happy to advise..

Problems? The Atari A—Team doesn't believe in them! With our efficient technical backup we can assist with any hardware or software queries.

So add experience, extensive catalogue and skilled technicians and it all equals the Atari A—Team a must for the consumer!

Now put us to the test.....

Yours sincerely

SOFTWARE EXPRESS

P.S. We can now offer an out of warranty repairs service!
Call our Service Hotline on (021) 350 9415.
P.P.S. Revision 'C' Basic -NOW AVAILABLE

TM ATARI CORPORATION LTD. TM COMMODORE (CBM) LTD.

NEW ATARI PACKAGES - POWER WITHOUT THE PRICE!

800XL + 1010 RECORDER + SOFTWARE = £129
800XL + 1050 DISK DRIVE + SOFTWARE = £269

Instead of just buying a 64K Atari 800XL for £129, why not buy it with a 1010 Program Recorder (RRP £34.99), Pole Position cassette (RRP £9.99) and Invitation to Programming (RRP £19.99) for the same price?! Or buy your Atari 800XL (RRP £129.99) with a 1050 Disk Drive (RRP £199.99), Home Filing Manager (RRP £24.99) and The Payoff (RRP £9.99) for only £269! That is what these two hardware packages offer you, the chance to buy a real value for money starter pack which will give you a good introduction to the world of computers. Furthermore, if you buy your pack from Silica Shop, you can be assured of after sales service and information, a FREE Owners Club and an extra 12 month guarantee on your 800XL computer, FREE OF CHARGE!

ATARI 800XL & 1010 RECORDER PACK	
64K Atari 800XL	£129.99
1010 Program Recorder	£34.99
Pole Position + Demo Prog (C)	£9.99
Invitation to Programming 1	£19.99
Total (if purchased separately)	£194.96
Saving	£65.96
PACKAGE PRICE (XLC 1010)	£129.00

ATARI 800XL & 1050 DISK DRIVE PACK	
64K Atari 800XL	£129.99
1050 Disk Drive	£199.99
Home Filing Man. (D)	£24.99
The Payoff + Demo Prog (D)	£9.99
Total (if purchased separately)	£364.96
Saving	£95.96
PACKAGE PRICE (XLC 1050)	£269.00

ORDER YOUR PACK NOW USING THE CUT OUT COUPON BELOW

UPGRADE YOUR OLD STYLE ATARI 800 TO AN 800XL & DISK DRIVE – ONLY £199

If you currently have an old style Atari 800 and want to purchase the new Atari 1050 Disk Drive, Silica are offering a unique upgrade facility. If you purchase a 1050 Disk Drive at £199 Silica will swop your old Atari 800 for a new 64K 800XL totally free of charge. In addition we will give you all of the constituent items in the Disk Drive Pack, i.e. a free Home Filing Manager program and the Payoff and Demo programs all on disk. The total price of the 800XL, 1050 and the programs if purchased seperately would be £364.96. You can have them all for the part exchange price of only £199. If you are interested in using our upgrade facilities, please send your old style Atari 800 to us (in working order and reasonable condition) along with £199 payment for the 800XL and Disk Drive pack. Providing we are satisfied with the condition of your old 800, we will despatch the 800XL and Disk Drive to you by return post and packing free. Please note that your new 800XL already has Basic built-in, so you should return your old Basic cartridge with the 800. In addition you should also return the Basic Reference Manual and the operators manual. You should however keep any programming books such as Atari Basic by Albrecht/Wiley or Inside Atari Basic by Carris. Use the order form below, to ensure a speedy service.

TOP ARCADE ROM TITLES - FROM ONLY £7.95!

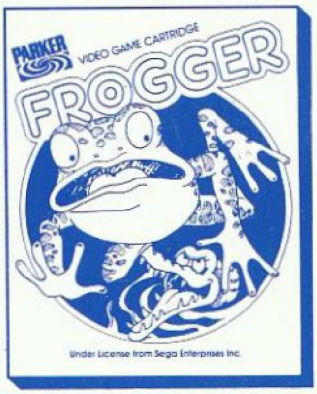
We are pleased to be able to offer you some top arcade titles on ROM cartridge at amazingly low prices. Previously priced at £29.95 each, the five cartridges we have on offer are all high quality licensed reproductions of top arcade games. Manufactured by Parker, FROGGER, Q*BERT, SUPER COBRA and POPEYE are all suitable for use with the new 600XL and 800XL machines as well as the old style Atari 400 and 800 computers. Astrochase however, will not run on the XL machines but is compatible with the old style Atari 400 and 800. Astrochase and Frogger are priced at £7.95 with Popeye, Q*bert and Super Cobra at £9.95. Brief game descriptions are given below and you may order cartridges using the coupon below. In addition to these special offers, we have a range of over 2,000 titles for Atari Computers, on Cassette, ROM Cartridge and Diskette, with prices starting at £1.99. Write, telephone, or complete the coupon for a copy of our latest price list.

FROGGER ROM £7.95

Home may look like it's only a hop, skip and a jump away, but looks can be deceiving. First, there's a dangerous highway to hop across, full of cars and trucks. Then there's a swirling river to leap, full of frog-eating creatures. How will frogger get home safely? By letting you help him on his way. Guide Frogger home safely through this perilous journey based on the highly successful arcade game. The object of the game is to hop as many frogs to safety as you can-scoring points along the way! Frogger's journey takes him from the sidewalk across a busy highway to the riverbank and across the river to a home bay and safety. You'll start the game with 5 frogs and the game ends when all frogs have been lost.

ASTROCHASE ROM £7.95

This is a fast flying race through space. Your mission is to save the earth from destruction. Blast off and blaze across a scrolling, star studded galaxy. Seek out and explode mega mines that close in on you as they orbit the earth. But keep an eye out for attacking enemy vessels, they'll pursue you to the far corners of space! Fire your lasers or activate your shields and ram the enemy. You have got single thrust propulsion which is an amazing feature which lets you fire in one direction whilst speeding off in another. Save the earth from total doom and there is a secret reward waiting for you. This game gives spectacular graphics and requires both craft and strategy if you are to succeed in your mission.



POPEYE ROM £9.95

You are Popeye and your goal is to catch all of Olive's hearts, notes, and cries for Help before they hit the water and sink - or before Brutus, the Sea Hag or a Vulture knock you overboard. Each time you complete a round, you will automatically proceed to the next and more difficult level. Popeye can gain strength during the game by eating spinach. He is then able to knock Brutus overboard for bonus points. An accurate reproduction of the popular arcade game, Popeye is for 1 or 2 players.

Q*BERT ROM £9.95

The object of Q*bert is to score as many points as possible by changing the colour of the cubes on a pyramid from a starting colour to a destination colour. You'll do this by hopping Q*bert from cube to cube while avoiding the nasty characters who will try to stop him. A faithful reproduction of the original, Q*bert requires nerves of steel!

SUPER COBRA ROM £9.95

Based on a top arcade game, Super Cobra provides a test for the most skilled and daring player. Guide your helicopter gunship through 10,000 miles of mountains, skyscrapers and subterranean passages. Only then can you pick up the booty and make your getaway. You'll have missiles, coming at you from all directions, determined to shoot you down. The course comprises 11 sections of terrain, each more difficult than the last.

SILICA SHOP ARE THE UK's No1 ATARI SPECIALIST

WE ARE THE UK'S No1 ATARI SPECIALISTS
Since the introduction of Atari Video Games into the UK six years ago, we have been selling Atari products and supporting Atari owners with a specialist mailing service which we believe is unbeatable. We stock over 1,000 Atari related product lines and have a mailing list including over 300,000 Atari 2600 VCS owners and over 50,000 Atari Home Computer Owners. Because we specialise in Atari, we aim to keep stocks of all available Atari hardware, software, peripherals and accessories. We also stock a wide range of Atari dedicated books and through us, the owners on our list can subscribe to several American Atari dedicated magazines. We can provide a full service to all Atari owners and are now firmly established as the UK's No1 Atari specialists.

RECOGNISED ATARI SERVICE CENTRE
We are an approved Atari Service Centre and able to repair any Atari equipment on site, no matter where it was purchased. This means that if your machine develops a fault, we can normally repair it ourselves, without having to send it away and perhaps wait weeks for its return. On average, repairs are carried out within five days (including continuous soak testing), so customers are without their machine for the shortest possible time. We have competitive prices for repairs on machines which are out of the warranty period (remember you get an extended 2 year guarantee on your new Atari 800XL if you buy it from Silica Shop). All repair work carries a 3 month guarantee.

FREE POST AND PACKING ON MAIL ORDERS
There is no charge for delivery within the UK and BFPO. Small packets are despatched by 1st class post and larger items by the Post Office or other national carriers. Approximately 80% of all the orders sent to the Silica Shop mail order department at Sidcup are processed through our computer and despatched to the customer on the same day that the order is recieved by us. This service is provided FREE OF CHARGE.

FREE OWNERS CLUB & INFORMATION SERVICE
Service is a keyword at Silica and therefore we are keen NOT to make you the subject of a 'quick sale' with no subsequent support. We would rather continue to communicate with you via our mailing facilities. When you make a purchase from Silica, your name and address will be taken and added to one of our FREE computer or video game clubs. We don't limit our after sales services only to those people who bought their hardware from us, our doors are open. So, if you own a video game or home computer, or are interested in buying one, do let us know.

YOU GET THE BEST PRICES AT SILICA SHOP
With an annual turnover of £2 million, we are often able to keep our buying prices low by bulk purchasing and it is our policy to pass on part of these savings to our customers. Because of this, we believe our prices to be lower than any of our competitors. However, if you should find a better offer, please contact us. We will aim to beat that price and in accordance with our usual policy, send the goods to you post and packing free. For full details of our credit facilities and a written quotation, please contact our sales office.

SILICA SHOP LTD, 1-4 The Mews, Hatherley Road, Sidcup, Kent, DA14 4DX Tel: 01-309 1111

ORDER NOW-OR JOIN OUR FREE ATARI OWNERS CLUB

To: SILICA SHOP LTD, Dept ATCOC 0485, 1-4 The Mews, Hatherley Road, Sidcup, Kent, DA14 4DX Telephone: 01-309 1111

LITERATURE REQUEST:

☐ Please enrol me in the FREE Silica Atari Owners Club and send me your FREE brochures.

☐ I own a ..... Videogame ☐ I own a ..... Computer

Mr/Mrs/Ms: ..... Initials: ..... Surname .....

Address: .....

..... Postcode: .....

ORDER REQUEST:

PLEASE SEND ME:

☐	800XL & 1010 Recorder Pack £129	☐	Frogger ROM Cartridge £7.95
☐	800XL & Disk Drive Pack £269	☐	Astrochase ROM (400/800 only - not XL) £7.95
☐	800XL & Disk Drive Upgrade	☐	Popeye ROM Cartridge £9.95
	(I enclose my old style 800, complete & working) £199	☐	Q*Bert ROM Cartridge £9.95
		☐	Super Cobra ROM Cartridge £9.95

ALL PRICES QUOTED ARE INCLUSIVE OF VAT - POSTAGE & PACKING IS FREE OF CHARGE

☐ I enclose Cheque/P.O. payable to Silica Shop Limited for the following amount £ .....

☐ CREDIT CARD - Please debit my:
Access/Barclaycard/Visa/American Express/Diners Club Card Number .....

01-309 1111