The U.K. ATARI Computer Owners Club Issue 18 Price £1.00

Independent User Group

# Monitor

Inside this Issue
GEM programming from C
ST routines in Assembler
6502 machine code

ST Reviews
Rampage
DXpert
Mailshot Plus
Chessbase
Super Sprint

8-Bit Reviews
Music Matrix
Storm
XL/XE Mouse
Amourote

XL/XE and ST

# FREELANCE ST ARTIST OR PROGRAMMERS

Group Dynamics are looking for people who can create high quality pictures, using Degas, for backgrounds, company logos, etc., and also for C and 68000 programmers to support our electronic voting system graphics.

We aren't looking for full time employees, nor, indeed, do you need to work in our offices; so if you already draw, or program for a hobby, then why not supplement your income?

If you are interested, drop us a line at Group Dynamics or call Richard Green on 01 580 3767 so that we can organise a chat and look at your work.

# CONFERENCE TECHNICIANS

## (Salary Negotiable)

European leader in the specialist field of conference voting technology, Group Dynamics, urgently needs several conference technicians, due to very rapid expansion.

Your responsibilities will be to install and run electronic keypad voting systems in venues throughout Europe.

You will initially work under a team leader, but in the very near future you will be running your own conference group. You will be expected to ensure that all hardware and software needed for an event operates to the full satisfaction of our clients, who are all major blue chip multinationals.

You will be in your early twenties, and of graduate calibre. You will certainly have a good understanding of electronics, and be able to program in C, and, ideally 68000 Assembler.

You won't be frightened of taking responsibility and assuming the lead position. Because you will travel extensively in the U.K., but predominately in Europe, often as the person in charge, you will be able to think on your feet and take sensible decisions quickly and independently.

The package we offer is made up of a good salary plus bonus, and four weeks' holiday.

The position is an excellent opportunity to develop your leadership skills, enjoy using your programming and electronic abilities, and a chance to travel much of Europe working with a young, informal and relaxed company.

If you are interested, call 01 580 3766 for an application form or send your C.V. to Group Dynamics at the address below.

**GROUP DYNAMICS**
**46 Oxford Street, London W1N 9FJ**

## Dates to Remember

Make a note in your diary of the dates of the next two Atari shows. Both are going to be at a new venue, this time it's the Alexandra Palace in north London. The dates are April 22nd to 24th inclusive and November 25th to 27th inclusive. Once again they are being organised by Database Exhibitions, who did such a good job of the last show at the Novatel in Hammersmith. Alexandra Palace is a bigger venue than the Novatel, and as the number of exhibitors grows it makes sense to find a more suitable location. The two floor situation at the Novatel is generally considered to be unsatisfactory and this is another good reason to move. Maybe one day Database might consider holding an exhibition outside of London too, we are sure it would be as well attended as the 'southern' venue. How about it Database?

Another date to look out for is the 14th to 18th of September. This is the date set aside for the 1988 Personel Computer World Show at which Atari usually has its 'village'. This show is also moving, this time from Olympia to Earls Court just down the road. At this point in time it seems that Atari are considering whether to be there or not. The floor plan at Earls Court does not lend itself easily to a sectioned off Atari World such as existed at Olympia and it is possible that they may not attend, or if they do there may not be a specified Atari area. However, it is early days and things could well change, let's hope they do!

## Libraries

It appears that many people are unsure about whether they can use the PD libraries or not, both on the 8 bit and the ST. This applies particularly to new club members/subscribers. Let's see if we can clarify the situation. First a 'club membership' is the same thing as a 'subscription', i.e. as long as you have paid £5 (or £8 or £12 if you live outside the U.K.) for an annual subscription to the club, and that subscription is still valid, then you can use the libraries (either or both) when ever you wish. All you need do is read the rules at the top of the relivent library page and go from there. A list of library titles is also available on request, also see the library page. In addition you can send programs for inclusion in the library, all are welcome.

### CREDITS

| | |
|---|---|
| Editor | Roy Smith |
| Art Editor | Greg Buckley |
| Technical Editor | Ron Levy |
| Technical Editor | Keith Mayhew |
| Adventure Editor | Steve Hillen |
| ST Librarian | Mike Stringer |

# CONTENTS

Cover: Obliterator from Psygnosis.
Club Address: P.O. Box 3, Rayleigh, Essex SS6 8LR.

### ADVERTISEMENTS

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

# CRACKING THE CODE

## Part Fourteen
## by Keith Mayhew

We start our tour of the Operating System (OS) with the Central Input Output (CIO) utility. CIO provides a consistent method of writing and reading data to and from devices such as the keyboard, disk drive, cassette or display.

### Overview of the Operating System

Before we delve into the mysteries of the CIO it is useful to gain an insight into the basic structure of the OS and the services it offers.

The CIO is at the top of a hierarchy of routines used to support input and output. There is one device driver routine for each physical type of device, these are either permanent or can be loaded at any time for additional devices, such as RS232 ports. To support devices which are attached via the serial port, such as a disk drive, the Serial Input Output (SIO) utility provides a high-level communications facility.

The parts of the OS which are not directly involved in input or output operations are actually rather minimal. They are the interrupt services and the so-called monitor which is executed at power-up and reset to initialise the system and pass control to another piece of software, such as BASIC.

The floating point package (FPP), although part of the OS, is completely separate in its operation and is best thought of as a set of utility routines for your use in manipulating non-integer numbers.

The OS occupies the 8K bytes from E000 to FFFF in all systems, with the systems character set data taking the first 1K bytes from E000 to E3FF. The FPP resides adjacent to the OS in the 2K region from D800 to DFFF.

The XL & XE machines also have ROM in the 4K region from C000 to CFFF providing an additional character set from CC00 to CFFF. The extra ROM space has been used for the self-test procedure and other minor additions.

### RAM Usage

As you are no doubt aware, page zero of memory from 00 to FF is an important area of memory for any program using the 6502: it gives greater speed, smaller code size and, most significantly, the ability to perform indirection, i.e. a pair of locations can be used as a pointer.

The OS allocates the first 128 bytes of page zero for its own purposes leaving the other half from 80 to FF free for applications. However, if the floating point package is used, it makes active use of locations D4 to FF, leaving only 80 to D3 free. If you do not call any FPP routines then you are of course free to use them as you wish.

If a language such as BASIC is being run then it takes further locations. In the case of standard ATARI BASIC the following locations are used: 80 to CA and D2 to D3. This leaves a user's program running under BASIC with access to locations CB to D1, as BASIC also uses the FPP. That gives just seven bytes which are safe to access!!!

A program, such as a game, which never uses the OS can use all the memory as it wishes. Other programs will have to be very careful otherwise they may cause unexpected results!

Page one is always reserved for the stack, as dictated by the design of the 6502. If you plan to push a large number values onto the stack then you should either test that the stack does not overflow (wrap around on itself) or set up your own private stack using a page zero pointer.

The memory from pages two to six inclusive are considered reserved and so programs should load from page seven onward.

Locations 200 to 47F are used by the OS leaving locations 480 to 6FF free for user's data. Well not quite! The FPP reserves the locations from 57E to 5FF. This leaves page six as the only completely unused page in lower memory.

### OS Vectors

The ATARI operating system makes extensive use of vectors for access to its resources. A vector being a pointer to a routine to be executed. By using this method, it means that only the vectors need be kept at fixed locations while the actual routines can reside anywhere. By adhering to the use of these vectors and other pointers means that programs will work under different revisions of the operating system.

The pointers in RAM are particularly useful as they can be 'redirected' to your own routines, allowing you to extend or modify the functions of the OS.

### Central Input Output Utilities

The CIO is the most useful part of the IO system of the ATARI as it provides a consistent, mostly device independent, interface to any IO device. This means that to access different devices you do not need to know too much about the characteristics of any particular device.

In fact, the CIO access method is essentially directly modelled in BASIC's file access commands such as OPEN, CLOSE, INPUT, PRINT, GET and PUT. All BASIC does is translate these commands into calls to the CIO.

### Calling the CIO

All CIO calls are made via one vector at E456. The instruction at this address is a jump instruction to the actual CIO entry point. The data structures used to communicate between the CIO and the user's program are called IO Control Blocks. There are eight IOCBs of sixteen bytes each starting at 340 and extending to 3BF.

Each of these IOCBs hold information for communication via a 'channel'. Each channel is either open or closed and when open is performing either input, output or both to one and only one device. When one of the eight channels is closed any remaining data is transferred and the channel then becomes free for future use. Note that any channel can be assigned to any device: there is no difference between any of the eight channels.

Table 1 shows the allocation of the bytes within an IOCB and their descriptions. The most important of these are as follows: The function you wish the CIO to execute next for the appropriate channel is passed in ICCMD; Table 2 lists the available commands. ICSTA is the status of the last CIO command executed for the channel; Table 3 lists the status values. The user buffer which is used to hold the bytes to be transferred is pointed to

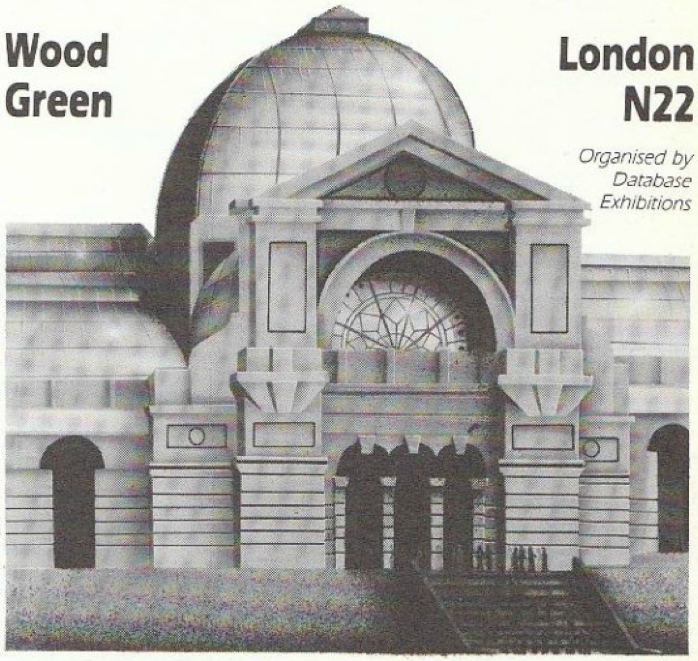
# *The phenomenal growth in demand for Atari computers means a much bigger home for the BIG show . . .*

# ATARI USER SHOW

| Friday | Saturday | Sunday |
|---|---|---|
| 22 April | 23 April | 24 April |
| 10am-6pm | 10am-6pm | 10am-4pm |

## ***THREE action-packed days for Atari owners!***

# Alexandra Palace



**Wood Green**

**London N22**

*Organised by Database Exhibitions*

**The spectacular Alexandra Palace represents a new showcase for the fastest-growing range of computers on the market.**

**From the exciting games console to the ever-popular 8-bit Atari and the sensational Mega ST, they'll all be there at the Atari User Show!**

**Some of the very latest software will be on show for the first time.**

**And that means you can enjoy unique hands-on experience of programs everyone will soon be talking about.**

**You'll find some of the best prices around for blank discs, disc boxes and other accessories, learn about the many opportunities to expand your computer system, get helpful advice from some of Britain's leading experts, and so much more!**

### *How to get there*

**It's so easy to get to the show – by car, rail, underground or bus. Alexandra Palace has its own British Rail station, whisking visitors to and from King's Cross in just nine minutes. And there's a free bus service shuttling between station and show every 10 minutes. If you're travelling by road the show is only 15 minutes away from junction 25 on the M25 – and all car parking is free.**

**Cut the queues and save £1 per head – with this advance ticket order**

### Your advance ticket order

*Please supply:*

□ Adult tickets at £2 (save £1) .............................. £..........
*(Order four adult tickets, get the fifth FREE!)*

□ Under-16s tickets at £1 (save £1).................... £..........
*(Order four under-16s tickets, get the fifth FREE!)*

Total £_____

□ I enclose a cheque made payable to Database Exhibitions

□ Please debit my Access/Visa card no:

Expiry date /

**_Admission at door:_ _£3 (adults),_ _£2 (under 16s)_**

**Advance ticket orders must be received by Wednesday, April 13.**

ATARI USER SHOW

**The West Hall**
**Alexandra Palace, Alexandra Park**
**Wood Green, London N22**
**April 22-24, 1988**

**Post to: _Atari User Show Tickets,_**
***Europa House, Adlington Park,***
***Adlington, Macclesfield SK10 5NP.***

Name.................................................................

Address..............................................................

.......................................................................

Postcode........................ Signed..........................

**PHONE ORDERS: Ring Show Hotline: 0625 879920**
**PRESTEL ORDERS: KEY *89, THEN 614568383**
**MICROLINK ORDERS: MAILBOX 72:MAG001**
*Please quote credit card number and full address*

A249
2.88(23)

## IOCB Locations

| Location | Name | Description |
|---|---|---|
| IOCB+0 | ICHID* | Handler ID |
| IOCB+1 | ICDNO* | Device Number |
| IOCB+2 | ICCMD | Command |
| IOCB+3 | ICSTA | Status. Same as Y register after call |
| IOCB+4 | ICBAL | Buffer address low & high |
| IOCB+5 | ICBAH | |
| IOCB+6 | ICPTL* | 'Put' address |
| IOCB+7 | ICPTH* | |
| IOCB+8 | ICBLL | Buffer length low & high |
| IOCB+9 | ICBLH | |
| IOCB+A | ICAX1 | Auxiliary bytes |
| IOCB+B | ICAX2 | |
| IOCB+C | ICAX3 | |
| IOCB+D | ICAX4 | |
| IOCB+E | ICAX5 | |
| IOCB+F | ICAX6 | |

* Intended for internal use only.

Table 1. IOCB Locations

## CIO Command bytes

| Value | Operation |
|---|---|
| 03 | Open channel |
| 05 | Get record |
| 07 | Get characters |
| 09 | Put record |
| 0B | Put characters |
| 0C | Close channel |
| 0D | Get status |
| 11 | Fill area |
| 12 | Draw line |
| 20 | Rename file |
| 21 | Delete file |
| 22 | Format disk |
| 23 | Lock file |
| 24 | Unlock file |
| 25 | Point in file |
| 26 | Note from file |

Note: values 11 and 12 are for display only, and values 20 to 26 are for disk only.

Table 2. CIO Command Bytes

## CIO Error Codes

| Hex | Decimal | Description |
|---|---|---|
| 1 | 1 | Operation successful (no errors) |
| 80 | 128 | Break key abort |
| 81 | 129 | IOCB already open |
| 82 | 130 | Non-existent device name |
| 83 | 131 | Opened for writing only |
| 84 | 132 | Invalid command byte |
| 85 | 133 | Device/file not open |
| 86 | 134 | Invalid IOCB channel specified in Y register |
| 87 | 135 | Opened for reading only |
| 88 | 136 | End of file reached on read |
| 89 | 137 | Truncated record |
| 8A | 138 | Device timeout |
| 8B | 139 | Device NAK (negative acknowledge) |
| 8C | 140 | Serial bus input framing error |
| 8D | 141 | Cursor out of range |
| 8E | 142 | Serial bus data frame overrun error |
| 8F | 143 | Serial bus data frame checksum error |
| 90 | 144 | Device 'done' error |
| 91 | 145 | Invalid screen mode |
| 92 | 146 | Function not supported |
| 93 | 147 | Insufficient memory for screen mode |
| A0 | 160 | Disk drive number invalid |
| A1 | 161 | Too many open disk files |
| A2 | 162 | Disk full |
| A3 | 163 | Fatal disk IO error |
| A4 | 164 | Internal file numbers inconsistent |
| A5 | 165 | File name invalid |
| A6 | 166 | 'Point' to non-existent sector |
| A7 | 167 | File 'locked' |
| A8 | 168 | Command invalid for DOS |
| A9 | 169 | Directory full |
| AA | 170 | File not found |
| AB | 171 | 'Point' to position out of file |

Table 3. CIO Error Codes

by ICBAL & ICBAH, its length is held in ICBLL & ICBLH. ICAX1 through ICAX6 hold device dependent information.

The lists in Table 2 and Table 3 are the standard values which are implemented on all systems and include the basic disk operations which all versions of DOS support. New device drivers may add commands and new error codes to those given: refer to the appropriate manuals for further details.

To execute any CIO command the following needs to be done:

1. Fill in any necessary bytes in an IOCB.
2. Load the X register with the index of this IOCB from the location 340 hex. This is equivalent to 16 times the channel number (0 to 7).
3. Jump to subroutine at E456 hex.
4. If necessary, examine the Y register, which holds a copy of the status code stored in the IOCB, to determine if an error has occurred. Note that all errors have bit 7 set, i.e. they are negative, so a BMI instruction will detect this upon return from CIO.

## Basic CIO Commands

The first seven commands in Table 2 are the most common CIO functions which are supported by almost every device. A description of each follows:

### OPEN

The OPEN IOCB command (03) has to be performed before most other CIO commands can take place on any particular IOCB.

The buffer address (ICBAL & ICBAH) must point at an ASCII string terminated with an end-of-line (EOL) character of 155 decimal (EOL is also the character returned by pressing the 'RETURN' key).

The ASCII string consists of an initial letter specifying the device, an optional number to specify one of several of those devices, e.g. two disk drives, a colon (:) and an optional filename for that device, e.g. a disk. This is in fact exactly the same as the names which are used in a BASIC filename. The following are the standard device names:

E: Editor.
S: Screen.
K: Keyboard.
C: Cassette.
P: Printer.
D: Disk drive. Not resident.
R: RS232. Not resident.

Note that the disk, i.e. DOS, and the RS232 devices are not resident in the computer and should be added to the system during power-up if they are required. It is also important to note that the editor is sometimes opened by the OS so that it can display messages or to prepare the screen ready for another application. IOCB number zero is always reserved by the OS for this purpose and it is best not to close and open it for devices other than 'E:'.

ICAX1 & ICAX2 are used to convey (device dependent) information on how to open the specified device. These two locations correspond to the numbers specified in a BASIC OPEN command. Usually zero will be placed in ICAX2 and ICAX1 will have 4 for read-only, 8 for write-only and 12 for read & write. ICAX3 through ICAX6 are used by very few devices and can be ignored in most cases.

### CLOSE

The CLOSE IOCB command (0C) is used on an IOCB which has previously been opened and is no longer required. No extra information needs to be specified (not even a filename). A close will ensure that any remaining data is written to a device.

### GET & PUT Characters

The GET characters command (07) reads the number of bytes specified by the buffer length (ICBLL & ICBLH) into the memory specified by the buffer address (ICBAL & ICBAH). The PUT characters command (0B) writes the specified number of bytes from the buffer to a device.

For either command a zero buffer length can be specified in which case the character is read into or written from the 6502's accumulator. All IOCB values are left as they were before the call except that a GET characters command returns the number of bytes actually read in the buffer length locations. If fewer bytes have been read then it is because an end-of-file (EOF) has been encountered.

### GET & PUT a Record

These two commands are similar to the previous two commands but these are useful when you do not know in advance exactly how many characters you wish to read or write. The GET record command (05) reads bytes into the specified buffer until an EOL character is encountered and written to the buffer. You still have to specify how long the buffer is and get the number of bytes actually read returned in ICBLL & ICBLH. If the buffer fills before an EOL is found then CIO reads the remaining characters from the device (ready for the next read) and returns an error code for a truncated record.

The PUT record command (09) writes all the characters in the specified buffer up to and including the first EOL character to the device. If the buffer does not contain an EOL then CIO will automatically write one to the device afterwards.

### STATUS

All CIO calls return a status code in the Y register immediately after a call which can be tested to see if an error had occurred. In addition this value is saved in the status byte of the IOCB for later reference.

The STATUS command (0D) performs more extensive status information which is device dependent. However, all resident handlers do not support this function and simply return one in the Y register: the code for 'no error'.

## An Example Program

Listing 1 is the assembly language code for a routine which prompts the user for an input and an output device/filename and copies from one to the other until end-of-file is reached. This performs exactly the same operation as the copy utility in DOS but is useful to illustrate how most of the CIO works and thus how to use it in your own programs. Listing 2 is the BASIC program to read the code into memory which is executed by typing:

X=USR(24576)

The program has several routines which provide the interface with the CIO routines. Each expects certain IOCB parameters to be set and the X register to contain the IOCB index, i.e. the IOCB channel number times sixteen. 'OPEN' opens an IOCB but closes it first to ensure that it will be free. It assumes that the buffer address has been set to a filename string and that the auxiliary bytes have been set appropriately. 'READLN' reads a record into the specified buffer. 'WRITELN' writes a record from the specified buffer. The buffer length is always set to FFFF as the record should be followed by an EOL. 'GETBYTE' reads a single character into the accumulator. No IOCB parameters need to be set. 'PUTBYTE' writes a single character from the accumulator. No IOCB parameters need to be set. 'GETCHRS' reads the specified number of characters into the given buffer. 'PUTCHRS' writes the specified number of characters from the given buffer. 'CLOSE' closes the IOCB. No IOCB parameters need to be set.

The program starts by re-opening the editor as the program cannot guarantee that it will always be open. It then prints a title message and prints a prompt for the input filename. A call is made to 'INPUT' which reads a line of text typed at the keyboard via 'READLN' which will be terminated with the EOL character, i.e. return.

A test is then made to see if 'ERROR' needs to be called. This may be necessary in case the user pressed the break key which causes a CIO error. The specified filename is then opened for read-only and the process is repeated for the destination filename.

If the two files opened successfully then the file is copied by the routine 'CPYFILE'. This tries to read a buffer full of data from the first file. It then checks the status code. If it is not an error or it is end-of-file then it copies the buffer length, which indicates how many characters were actually read into the other IOCB and writes out the data.

After a write a further check is made to see if the previous read was at end-of-file. If it was not it jumps back to continue reading otherwise it returns to the user.

Any call to 'ERROR' calls on 'CIOERR' and then tries to close both the input and the output files. It then terminates the program by returning to the user. 'CIOERR' is a useful piece of code which prints an error message and the error code in decimal. Printing the code in decimal is not as easy as it may at first seem: the obvious way is to divide by multiples of 10 and to take the remainders. This is however a rather messy solution and slow if you wanted to extend it to big numbers.

The solution adopted is quite elegant: it sets decimal mode with 'SED' and then shifts the number to be converted one bit at a time to the left. On each shift the result bytes are doubled by adding each of them to itself, the first byte gets incremented if the carry was set from the shift operation. At the end of the loop, decimal mode is cancelled and the result contains the binary coded decimal (BCD) for the number. A simple printing routine then outputs each digit. A check is also made to suppress any leading zeros from the number to make the output neater.

It would be very easy to modify the

```
0100 ;Example program using CIO to copy files.
0110 ;
0120 ;O.S. vectors...
0130 CIOV    =       $E456
0140 ;IOCB locations...
0150 ICCOM   =       $0342   ;Command code.
0160 ICBAL   =       $0344   ;Buffer address low.
0170 ICBAH   =       $0345   ;Buffer address high.
0180 ICBLL   =       $0348   ;Buffer length low.
0190 ICBLH   =       $0349   ;Buffer length high.
0200 ICAX1   =       $034A   ;Auxiliary 1.
0210 ICAX2   =       $034B   ;Auxiliary 2.
0220 ;IOCB commands...
0230 COPEN   =       $03     ;Open device.
0240 CGETREC =       $05     ;Get record.
0250 CGETCHR =       $07     ;Get characters.
0260 CPUTREC =       $09     ;Put record.
0270 CPUTCHR =       $0B     ;Put characters.
0280 CCLOSE  =       $0C     ;Close device.
0290 ;O.S. equates...
0300 RD      =       $04     ;Read from file.
0310 WR      =       $08     ;Write to file.
0320 EOL     =       $9B     ;End of line flag.
0330 EOF     =       $88     ;End of file error.
0340 ;Program equates.
0350 EDIOCB  =       $00     ;IOCB index for editor 'E:'.
0360 FMIOCB  =       $10     ;IOCB index for 'from' file.
0370 TOIOCB  =       $20     ;IOCB index for 'to' file.
0380 LNBUFL  =       $80     ;Length of line buffer.
0390 BUFLEN  =       $0400   ;Length of copy buffer.
0400 ;Page zero variables...
0410         *=      $CB
0420 STATUS  *=      *+1     ;Status of CIO command.
0430 ZERO    *=      *+1     ;Flag used when printing.
0440         *=      $6000
0450         PLA
0460         LDX     #EDIOCB ;Open editor in case closed.
0470         LDA     #EFILE&$FF ;File spec. pointer.
0480         STA     ICBAL,X
0490         LDA     #EFILE/256
0500         STA     ICBAH,X
0510         LDA     #RD+WR  ;Read and write.
0520         STA     ICAX1,X
0530         JSR     OPEN
0540         LDA     #MSTITLE&$FF ;Print title string.
0550         STA     ICBAL,X
0560         LDA     #MSTITLE/256
0570         STA     ICBAH,X
0580         JSR     WRITELN
0590         LDA     #EOL    ;Skip two lines.
0600         JSR     PUTBYTE
0610         JSR     PUTBYTE
0620         LDA     #MSFROM&$FF ;Message for 'from' file.
0630         STA     ICBAL,X
0640         LDA     #MSFROM/256
0650         STA     ICBAH,X
0660         JSR     WRITELN
0670         JSR     INPUT   ;Get input line.
0680         BMI     ERROR   ;Catch errors e.g. 'BREAK'.
0690         LDX     #FMIOCB ;Try to open given file name
0700         LDA     #RD     ;for reading.
0710         JSR     OPNFILE
```

```
0720         BMI     ERROR
0730         LDX     #EDIOCB ;Message for 'to' file.
0740         LDA     #MSTO&$FF
0750         STA     ICBAL,X
0760         LDA     #MSTO/256
0770         STA     ICBAH,X
0780         JSR     WRITELN
0790         JSR     INPUT
0800         BMI     ERROR
0810         LDX     #TOIOCB ;Try to open file.
0820         LDA     #WR
0830         JSR     OPNFILE
0840         BMI     ERROR
0850         JSR     CPYFILE ;Both files open: start copy.
0860         LDX     #FMIOCB ;Close both files.
0870         JSR     CLOSE
0880         LDX     #TOIOCB
0890         JSR     CLOSE
0900         RTS             ;All done...
0910 ;Handle a CIO error...
0920 ERROR   STY     STATUS  ;Save status code.
0930         JSR     CIOERR  ;Print error message.
0940         LDX     #FMIOCB ;Make sure both files
0950         JSR     CLOSE   ;are closed.
0960         LDX     #TOIOCB
0970         JSR     CLOSE
0980         RTS
0990 ;Copy 'from' file to 'to' file...
1000 CPYFILE LDX     #FMIOCB ;'from' file buffer address.
1010         LDA     #BUFFER&$FF
1020         STA     ICBAL,X
1030         LDA     #BUFFER/256
1040         STA     ICBAH,X
1050         LDA     #BUFLEN&$FF ;Set length.
1060         STA     ICBLL,X
1070         LDA     #BUFLEN/256
1080         STA     ICBLH,X
1090         LDX     #TOIOCB ;'to' file buffer address.
1100         LDA     #BUFFER&$FF
1110         STA     ICBAL,X
1120         LDA     #BUFFER/256
1130         STA     ICBAH,X
1140 READNXT LDX     #FMIOCB ;Read a block into buffer.
1150         JSR     GETCHRS
1160         STY     STATUS  ;Save status.
1170         BPL     WRTBUF  ;If OK then write buffer.
1180         CPY     #EOF    ;If not EOF then error.
1190         BNE     ERROR
1200 WRTBUF  LDY     #FMIOCB ;Copy length of block.
1210         LDX     #TOIOCB
1220         LDA     ICBLL,Y
1230         STA     ICBLL,X
1240         LDA     ICBLH,Y
1250         STA     ICBLH,X
1260         JSR     PUTCHRS ;Write block.
1270         BMI     ERROR
1280         LDA     STATUS  ;Stop if EOF from last read.
1290         BPL     READNXT ;Else do next block.
1300         RTS             ;All copied.
1310 ;Open file specified by input buffer.
1320 OPNFILE STA     ICAX1,X ;Save read/write info.
1330         LDA     #$00    ;Zero auxiliary 2.
```

```
1340         STA     ICAX2,X
1350         LDA     #LINBUF&$FF
1360         STA     ICBAL,X
1370         LDA     #LINBUF/256
1380         STA     ICBAH,X
1390         JSR     OPEN
1400         RTS
1410 ;Get user input into line buffer.
1420 INPUT   LDA     #LINBUF&$FF
1430         STA     ICBAL,X
1440         LDA     #LINBUF/256
1450         STA     ICBAH,X
1460         LDA     #LNBUFL&$FF
1470         STA     ICBLL,X
1480         LDA     #LNBUFL/256
1490         STA     ICBLH,X
1500         JSR     READLN
1510         RTS
1520 ;Open a file, closing IOCB first.
1530 OPEN    JSR     CLOSE
1540         LDA     #COPEN
1550         STA     ICCOM,X
1560         JSR     CIOV
1570         RTS
1580 ;Read up to end of line.
1590 READLN  LDA     #CGETREC
1600         STA     ICCOM,X
1610         JSR     CIOV
1620         RTS
1630 ;Write up to end of line.
1640 WRITELN LDA     #CPUTREC
1650         STA     ICCOM,X
1660         LDA     #$FF    ;Maximum buffer length
1670         STA     ICBLL,X ;as string is terminated
1680         STA     ICBLH,X ;by EOL character.
1690         JSR     CIOV
1700         RTS
1710 ;Get a single byte in accumulator.
1720 GETBYTE LDA     #CGETCHR
1730         STA     ICCOM,X
1740         LDA     #$00    ;Zero buffer length.
1750         STA     ICBLL,X
1760         STA     ICBLH,X
1770         JSR     CIOV
1780         RTS
1790 ;Put a single byte from accumulator.
1800 PUTBYTE PHA
1810         LDA     #CPUTCHR
1820         STA     ICCOM,X
1830         LDA     #$00    ;Zero buffer length.
1840         STA     ICBLL,X
1850         STA     ICBLH,X
1860         PLA
1870         JSR     CIOV
1880         RTS
1890 ;Get specified number of characters.
1900 GETCHRS LDA     #CGETCHR
1910         STA     ICCOM,X
1920         JSR     CIOV
1930         RTS
1940 ;Put specified number of characters.
1950 PUTCHRS LDA     #CPUTCHR
```

```
1960         STA     ICCOM,X
1970         JSR     CIOV
1980         RTS
1990 ;Close file.
2000 CLOSE   LDA     #CCLOSE
2010         STA     ICCOM,X
2020         JSR     CIOV
2030         RTS
2040 ;Print CIO error status.
2050 CIOERR  SED             ;Decimal mode.
2060         LDA     #$00    ;Zero result.
2070         STA     NUMBER
2080         STA     NUMBER+1
2090         LDX     #8      ;Eight bits.
2100 NEXTBIT ASL     STATUS  ;Get next bit in carry.
2110         LDY     #1      ;Double number adding carry.
2120 ADDLOOP LDA     NUMBER,Y
2130         ADC     NUMBER,Y
2140         STA     NUMBER,Y
2150         DEY
2160         BPL     ADDLOOP ;Next byte.
2170         DEX
2180         BNE     NEXTBIT ;Next bit.
2190         CLD             ;Back to binary mode.
2200         JSR     PRNTERR ;Print error message.
2210         RTS
2220 ;Print CIO error message and number.
2230 PRNTERR LDX     #EDIOCB ;Print message.
2240         LDA     #MSERROR&$FF
2250         STA     ICBAL,X
2260         LDA     #MSERROR/256
2270         STA     ICBAH,X
2280         LDA     #MSERRL&$FF
2290         STA     ICBLL,X
2300         LDA     #MSERRL/256
2310         STA     ICBLH,X
2320         JSR     PUTCHRS
2330         LDA     #$00    ;Clear 'zero' flag.
2340         STA     ZERO
2350         LDA     NUMBER  ;Start with third digit.
2360         AND     #$0F
2370         STA     ZERO    ;Save.
2380         BEQ     MDDIGIT ;Don't print if zero.
2390         JSR     PUTNUM
2400 MDDIGIT LDA     NUMBER+1 ;Second digit.
2410         JSR     GETHIGH
2420         TAY             ;Save.
2430         ORA     ZERO    ;OR to previous digit.
2440         STA     ZERO
2450         BEQ     LODIGIT ;Skip if a leading zero.
2460         TYA             ;Print digit.
2470         JSR     PUTNUM
2480 LODIGIT LDA     NUMBER+1 ;Print first digit.
2490         AND     #$0F
2500         JSR     PUTNUM
2510         LDA     #EOL    ;Print end of line.
2520         JSR     PUTBYTE
2530         RTS
2540 ;Return high nibble of 'A' in low nibble.
2550 GETHIGH LSR     A
2560         LSR     A
2570         LSR     A
```

```
2580          LSR     A
2590          RTS
2600 ;Print a single digit.
2610 PUTNUM   CLC
2620          ADC     #'0'
2630          JSR     PUTBYTE
2640          RTS
2650 ;Messages...
2660 MSTITLE .BYTE   "COPY FILE",EOL
2670 MSFROM  .BYTE   "From file?",EOL
2680 MSTO    .BYTE   "To file?",EOL
2690 MSERROR .BYTE   "Error number "
2700 MSERRL  =       *-MSERROR ;Length of message.
2710 EFILE   .BYTE   "E:",EOL  ;Editor file spec.
2720 NUMBER  *=      *+2       ;Error number in decimal.
2730 LINBUF  *=      *+LNBUFL  ;Line buffer.
2740 BUFFER  *=      *+BUFLEN  ;Copy buffer.
```

Listing 1.

```
10 DIM HEX$(16)
20 LINE=10000:TRAP 100:J=0:START=24576
30 READ HEX$,CHKSUM:SUM=0
40 FOR I=1 TO 15 STEP 2
50 D1=ASC(HEX$(I,I))-48:D2=ASC(HEX$(I+1,I+1))-48
60 NUM=((D1-7*(D1>16))*16+(D2-7*(D2>16)))
70 SUM=SUM+NUM:POKE START+J,NUM:J=J+1:NEXT I
80 IF SUM=CHKSUM THEN LINE=LINE+10:GOTO 30
90 ? "Checksum error on this line:"
95 LIST LINE:END
100 PRINT "Data in memory."
10000 DATA 68A200A9FE9D4403,917
10010 DATA A9619D4503A90C9D,833
10020 DATA 4A0320F760A9D39D,989
10030 DATA 4403A9619D450320,598
10040 DATA 0C61A99B202E6120,640
10050 DATA 2E61A9DD9D4403A9,930
10060 DATA 619D4503200C6120,499
10070 DATA DF603034A210A904,770
10080 DATA 20C960302BA200A9,751
10090 DATA E89D4403A9619D45,952
10100 DATA 03200C6120DF6030,543
10110 DATA 17A220A90820C960,723
10120 DATA 300E208060A21020,528
10130 DATA 5361A22020536160,682
10140 DATA 84CB205C61A21020,766
10150 DATA 5361A22020536160,682
10160 DATA A210A9839D4403A9,875
10170 DATA 629D4503A9009D48,725
10180 DATA 03A9049D4903A220,603
10190 DATA A9839D4403A9629D,952
10200 DATA 4503A21020416184,576
10210 DATA CB1004C088D0C1A0,1112
10220 DATA 10A220B948039D48,699
10230 DATA 03B949039D490320,529
10240 DATA 4A6130ACA5CB10DA,993
10250 DATA 609D4A03A9009D4B,731
10260 DATA 03A9039D4403A962,670
10270 DATA 9D450320F76060A9,869
10280 DATA 039D4403A9629D45,724
10290 DATA 03A9809D4803A900,701
10300 DATA 9D49032003616020,493
10310 DATA 5361A9039D420320,610
10320 DATA 56E460A9059D4203,810
10330 DATA 2056E460A9099D42,843
10340 DATA 03A9FF9D48039D49,889
10350 DATA 032056E460A9079D,778
10360 DATA 4203A9009D48039D,627
10370 DATA 49032056E46048A9,759
10380 DATA 0B9D4203A9009D48,635
10390 DATA 039D4903682056E4,686
10400 DATA 60A9079D42032056,616
10410 DATA E460A90B9D420320,762
10420 DATA 56E460A90C9D4203,817
10430 DATA 2056E460F8A9008D,1000
10440 DATA 01628D0262A20806,516
10450 DATA CBA001B901627901,770
10460 DATA 629901628810F4CA,948
10470 DATA D0EDD8207F6160A2,1175
10480 DATA 00A9F19D4403A961,904
10490 DATA 9D4503A90D9D4803,643
10500 DATA A9009D4903204A61,605
10510 DATA A90085CCAD016229,819
10520 DATA 0F85CCF00320CC61,928
10530 DATA AD026220C761A805,774
10540 DATA CC85CCF0049820CC,1173
10550 DATA 61AD0262290F20CC,662
10560 DATA 61A99B202E61604A,766
10570 DATA 4A4A4A6018693020,527
10580 DATA 2E6160434F505920,586
10590 DATA 46494C459B46726F,738
10600 DATA 6D2066696C653F9B,775
10610 DATA 546F2066696C653F,706
10620 DATA 9B4572726F72206E,819
10630 DATA 756D62657220453A,698
10640 DATA 9B,155
```

Listing 2.

decimal conversion routine for larger numbers making it of more general use for other programs, e.g. printing memory address.

The error handling is rather primitive in this example: it always terminates the program. It is usual for a program to effectively 'trap' its errors causing some recovery action to be taken. For example, a writing error might ask the user to check the disk (if it was a disk drive) so that a retry can be made.

## Next Time

That completes the description of our example CIO program which should help you to write your own routines to perform general input or output. Next time we will look at how the CIO uses device drivers to implement the actual input or output to a device and the facilities which each of them offers.

# CHECKER

## By Gordon Cameron

Basic checker is a program which makes debugging of your basic programs a lot easier. How many times has your program refused to run as planned, and thrown up a cryptic error message number on the screen? Who can understand and decipher these strange hieroglyphics? If this situation appears all too familiar, then Checker could make life a whole lot easier. When you have entered your program, be it your own, or a listing from a magazine, and are ready to seek out the almost inevitable errors, simply merge the Checker program with your own. Type RUN and the Checker module takes over. Any error that occurs in your program will be detected, and a full text explanation will be shown on screen, together with where it occurred, and its code number (for those of you used to the old ways or unbelieving!).

You can then correct the mistake and try again. When you are satisfied that there are no more bugs, type RUN and choose the 'Delete Module' option from the menu. The Checker module will then delete itself, leaving your own program intact, and 4 lines to delete yourself. It's as simple as that! There are a few things to remember. Firstly, remember not to use line 0 or lines greater than 30000, as these are the lines used by Checker. (Very few programs need to use these lines, so this is a very minor 'limitation'). Secondly, when de-bugging using Checker, make sure that your program doesn't contain any TRAP statements, as these will override Checker. And lastly, if you are reading data in your program, remember not to read past the end of your own data and into the text used by the Checker module. In other words, ensure you have enough data of your own to READ(!), and remember to use RESTORE, if needed. (This problem does not arise if you are reading numeric data, Checker will give you an error message alerting you of the possibility that you have insufficient data.) Follow these 3 simple 'rules' and you should have no problems. Checker should make de-bugging a lot easier, and eliminate the need to constantly look up those cryptic error numbers.

To use Checker is very simple. First, type in the program (the two code letters at the start of each line are for use with the typing check programs, KEYO or TYPO, if you are not using one of these then jüst ignore the codes). SAVE it to disk or cassette normally, just as a master copy. To create the working copy, simply LIST the program to disk, (LIST D:CHECKER) or to cassette(LIST C:). When you are ready to use the module, simply merge your program with it using the ENTER command, (e.g. ENTER D:CHECKER for disk, ENTER C: for cassette). Type RUN and follow the menu.

```
AK 0 DIM REPORT$(80):TRAP 30050:GOSUB 300
   01
XI 30000 END
VY 30001 REM ERROR CHECKER (C) G.C.'87
OK 30010 GRAPHICS 0:? CHR$(125)
WO 30015 POKE 752,1
KU 30020 COLOR 160:PLOT 1,1:DRAWTO 38,1:D
   RAWTO 38,20:DRAWTO 1,20:DRAWTO 1,1
MS 30030 POSITION 15,5:? "Select :"
VZ 30040 POSITION 7,8:? "Execute Program
   -  START ":POSITION 5,10:? "Delete thi
   s module -  OPTION "
XK 30041 KEYPRESS=PEEK(53279):IF KEYPRESS
   =6 THEN POKE 752,0:? CHR$(125):RETURN
GX 30042 IF KEYPRESS<>3 THEN 30041
BW 30043 GOTO 30300:REM DELETE MODULE
ZR 30049 END
UU 30050 REM DISPLAY THE ERROR
QM 30052 GRAPHICS 0:? CHR$(125):PRINT "Th
   e error is as follows :-"
MP 30053 ERRNUM=PEEK(195):ERRLINE=PEEK(18
   6)+256*PEEK(187)
AW 30054 RESTORE 30100+ERRNUM
FB 30055 REPORT$="":READ REPORT$:? REPORT
   $:?
VH 30056 ? "It occured at line number ";E
   RRLINE:? "and has code ";ERRNUM;"."
AL 30099 END
TJ 30100 REM ERROR CHECK
PD 30102 DATA Out of Memory
RQ 30103 DATA Value out of allowable rang
   e
TV 30104 DATA There are too many variable
   s
YZ 30105 DATA String error (e.g.out of DI
   M range)
BL 30106 DATA Insufficient Data for READ
OW 30107 DATA Illegal line reference numb
   er
KF 30108 DATA Can't store string numerica
   lly.Also   check prog.isn't reading CH
   ECKER data!
GA 30109 DATA Array/String dimension erro
   r
```

```
PN 30110 DATA Expression too large/Too ma
   ny GOSUB's
II 30111 DATA Number over/under flow or d
   ivision by zero error
ME 30112 DATA Line referénced is non-exis
   tent
TL 30113 DATA NEXT without FOR error
XX 30114 DATA Entered line is too large (
   >140)
MW 30115 DATA Line with GOSUB or FOR dele
   ted
GD 30116 DATA RETURN without GOSUB error
TL 30117 DATA Random garbage error
UF 30118 DATA Invalid character in string
   ,or non    numeric VAL expression
HP 30119 DATA Not enough memory for progr
   am loading
WJ 30120 DATA Device number error ( >7 or
    <0 )
ML 30121 DATA Load file error - must use
   ENTER or   CLOAD
ZX 30228 DATA Break pressed during I/O
LA 30229 DATA Cannot open channel - Alrea
   dy open !
KY 30230 DATA Non-specified device
KC 30231 DATA Cannot read to file open fo
   r write
PK 30232 DATA Illegal I/O command
OB 30233 DATA Channel referenced is not o
   pen
MZ 30234 DATA Illegal channel number ( <0
    or >7 )
RW 30235 DATA Cannot write to file open f
   or read
WK 30236 DATA End of file has been reache
   d
PE 30237 DATA Record too big or cannot us
   e INPUT    to access file which has be
   en PUT
NW 30238 DATA Device Timeout
DS 30239 DATA No response - Device NAK
XH 30240 DATA Serial frame error - faulty
    computer  or peripheral
OF 30241 DATA Cursor outside allowable ra
```

```
   nge
LV 30242 DATA Fault in serial bus
DD 30243 DATA Corrupted data from periphe
   ral
UM 30244 DATA Cannot write to write-prote
   cted disk
LX 30245 DATA Screen handler error
WC 30246 DATA Use of device not permissab
   le ( Not   Implemented )
AW 30247 DATA Insufficient RAM for Screen
    mode
NA 30260 DATA Incorrect drive number
YO 30261 DATA Too many files are open
RL 30262 DATA Disk is full
OR 30263 DATA Error in DOS/System I/O err
   or
DH 30264 DATA Incorrect use of POINT
UE 30265 DATA Illegal filename used
HO 30266 DATA Specified byte non-existent
   /POINT     error
KT 30267 DATA File is locked
AG 30268 DATA Illegal device specified
HE 30269 DATA Too many files on disk ( >6
   4 )
OR 30270 DATA Can't find specified file o
   n disk
MB 30271 DATA POINT error - File incorrec
   tly opened
VU 30272 DATA Illegal Append/wrong DOS
NB 30273 DATA Cannot format - Bad sectors
    on disk
UU 30300 REM DELETE MODULE
BO 30301 START=30000:LAST=30121:GOSUB 303
   05
LA 30302 START=0:LAST=0:GOSUB 30305
AK 30303 START=30228:LAST=30303
EB 30304 GOSUB 30305:POKE 842,14:? "NOW D
   ELETE LINES 30304,30305,30306 &   3030
   7..THANKS AND BYE..":POKE 752,0:END
SE 30305 FOR LINE=START TO LAST
RD 30306 ? CHR$(125):POSITION 2,2:? LINE:
   ? "GOTO 30307":POSITION 0,0:POKE 842,1
   3:END
CX 30307 NEXT LINE:RETURN
```

# EIGHT BIT SOFTWARE

## Software Librarian - Roy Smith

There are two ways to get programs from the library. You can use the donation scheme by sending in a disk or cassette of your own, or if you have a program of your own which you would like to add to the library you can exchange it for 3 programs of your choice. The rules are as follows:

### 3 FOR 1 EXCHANGE

1. Every program you donate entitles you to three programs in return.

2. The program you donate must be your own original and not copied.

3. Your donated program must be submitted on a cassette or a disk, programs in the form of print-outs cannot 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 submitted per quarter (between issues of the magazine) will be eligible to be judged 'STAR PROGRAM' for that quarter. This carries a prize of £10 which will be paid to the author. The programs will be judged by the Editorial Team and their decision is final. The Editorial Team are not eligible for the prize.

6. Please include 30p in stamps (or cash) to cover return postage.

7. The '3 for 1' exchange is only open to club members.

### DONATION SCHEME

1. Every club member can make a donation to the club, at any time, if he/she wishes to obtain a particular program(s).

2. There is no limit on the number of programs that can be asked for at any one time. (If you are asking for a lot of programs at once, please ensure that you send a sufficient number of disks or cassettes. It's better to send too many than not enough.)

3. Please include 30p in stamps (or cash) minimum to cover return postage. If your parcel costs more than 30p to send to us, please include an amount equal to that of the postage, so that we may return your parcel to you without delay. Overseas members should add an extra £1 to cover postage costs.

4. The donation fee is £1 per program. Cheques or Postal Orders should be made out to the 'U.K. Atari Computer Owners Club'.

5. You should send in blank disks or cassettes, ensuring they are properly packed to prevent damage in the post. State which programs you require and remember to give your name and address. Also remember to include the fee and return postage.

6. The 'Donation Scheme' is only open to club members.

**The Library Software Service is for subscribers only**

# *LIBRARY SOFTWARE TITLES*

Listed below are the software titles received by members for inclusion in the library since the last issue was published. As the library now contains over 350 programs, it is getting too big to print the entire list. For those of you who are new to Monitor and are unaware of what is available, then 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 send 50p and a S.A.E. for return.

## Games

### AQ

by R. Fake - Edgware.
Stocks and bonds trading game for up to four players.
Runs in 32K min. Disk only.
XL/XE only.

### KILLA CYCLE

by Evan Skinner - Stockport.
Two player game, who can last the longest? Snake type game.
Runs in 16K min. Disk or Cassette.

### MASTERMIND

by P. Sinclair - Crewe.
Guess the correct number to win, 9 levels of difficulty.
Runs in 16K min. Disk or Cassette.

### MULTI-MILLIONAIRE

by Reg Hatch - Datchet.
Can you make lots of cash and end up a multi- millionaire?
Runs in 64K min. Cassette only.
XL/XE machines only.

## Demos

### DIGI-TUNES

A selection of digitised tunes (well bits of anyway) including 19, Axel F and Into the Groove.
Runs in 48K min. Disk only.
Requires 1 side of a disk.
1050 ONLY. 400/800 ONLY.

### FANTASY PICS

Six highly colourful sci-fi and horror pictures.
Runs in 32K min. Disk only.
Requires 1 side of a disk.

### MAGIC LANTERN

by A. Lefreniere - Canada.
25 koala pad pics displayed one after the other.
Runs in 48K min. Disk only.
Requires 1 side of a disk.

### MATT HOUSTON'S GRAPHICS

9 digitised pictures, some in mono, some in colour.
Runs in 48K min. Disk only.
Requires 1 side of a disk.

### POP DEMO

Combination demo disk from Germany. Includes 256 colours demo, music and graphics demos.
Runs in 32K min. Disk only.
Requires 1 side of a disk.

### SHUTTLE

by J. Bean - Colwyn Bay.
Animated demo of Space Shuttle launch.
Runs in 32K min. Disk or Cassette.

## Utilities

★★★ **STAR PROGRAM** ★★★

### ACE BUSINESS DISK 4

by Ace of Eugene Oregon - USA.
Small business programs including Smartsheet and Checkbudget.
Runs in 48K min. Disk only.
Requires 1 side of a disk.

★★★★★★★★★★★★★★★★

### BASIC CHECKER

by Gordon Cameron - Scone.
Merge this program with your own programs and let it check for errors.
Runs in any size. Disk or Cassette.

### LISTER 1020

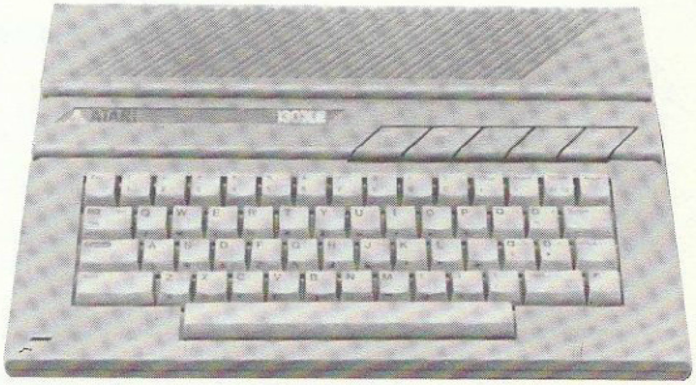
by Steven Burke - Bexleyheath.
Mini database for lists that can be printed on a 1020 plotter in 80 column mode. Includes forward and reverse scrolling and menu list selection.
Runs in 48K min. Disk only.

### RANDOM BIAS

by N. Edwards - Leighton Buzzard.
Generates random numbers, gives a graph and analysis of the results so you can tell if your computer is biased towards high or low numbers.
Runs in 48K min. Disk or Cassette.

# Ramdisk Mover

**By Trevor Peart-Jackson**
**For 130XE Only**



This short program will create an AUTORUN.SYS file which when used with DOS 2.5 and RAMDISK.COM on your 130XE will display the disk directory on boot up.

Single key presses (A, B, C, etc.) will select the files to be moved to RAMdisk. Once RETURN is pressed the selected files will be moved to RAMdisk and control passed to Basic. Should no files be required to be moved, just press RETURN for Basic. The occasional lock-up is cleared by pressing BREAK, with no memory damage.

The two letter codes at the beginning of each line are for use with a typing check program such as TYPO, or our own KEYO (see issue 9). If you do not use a typing checker program then just ignore these two codes.

```
SL 10 GRAPHICS 0:DIM CODE$(93),HEX(22),A$
   (1024)
LH 20 FOR I=0 TO 22:READ J:HEX(I)=J:NEXT
   I:? :? "RAMDISK MOVER ":OUTPUT=0:X=1:?
    :? "***CHECKING DATA"
GU 40 TRAP 110:RESTORE 1000:FOR LINE=1000
    TO 1200 STEP 10:? "LINE ";LINE;:TOTAL
   =0
WH 50 READ CODE$:IF LINE<>PEEK(183)+256*P
   EEK(184) THEN ? " MISSING!":END
VH 60 IF LEN(CODE$)<>92 THEN 100
BH 70 FOR I=1 TO 46:BYTE=16*HEX(ASC(CODE$
   (2*I-1))-48)+HEX(ASC(CODE$(2*I))-48)
QT 80 IF OUTPUT THEN PUT #1,BYTE:NEXT I:R
   EAD CHKSUM:? :NEXT LINE:CLOSE #1:? :?
   "***FILE CREATED":END
VW 90 TOTAL=TOTAL+I*BYTE:NEXT I:IF TOTAL>
   1000 THEN TOTAL=TOTAL-1000*(INT(TOTAL/
   1000))
RU 100 READ CHKSUM:IF TOTAL=CHKSUM THEN ?
    :NEXT LINE:GOTO 120
LF 110 ? " BAD DATA":END
SH 120 ? :? "***CREATE FILE":OUTPUT=1
ME 130 ? :? "WRITING TO DISK "
WP 135 OPEN #1,8,0,"D:AUTORUN.SYS"
QU 170 GOTO 40
MQ 180 DATA 0,1,2,3,4,5,6,7,8,9,0,0,0,0,0
   ,0,0,10,11,12,13,14,15
PL 1000 DATA FFFF0030B833A9008DA133A220A9
   039D4203A97B9D4403A9339D4503A9029D4A03
   2056E4A220A9079D4203A9009D,836
VZ 1010 DATA 4403A9349D4503A9009D4803A904
   9D49032056E42056332072 33A941BDA033A000
   A93191CBC8A93A91CBA00CB1CB,895
NE 1020 DATA C900F01DC891CBA00BB1CBC891CB
   A00AB1CBC891CBA00AA92E91CBEEA1334C7A30
   4C2631A000B1CB99A333C8C011,92
PK 1030 DATA F0034C7C30A010A92091CB88C0FF
   F0034C8B30A201A00AE8BDA333C920F007E00A
   F0034C9B30B9A3339DA333C8E8,952
ZE 1040 DATA C00EF0034CAA30A9209DA333E8E0
   11F0034CBB30A211BDA333C920D004CA4CC830
   C92EF0034CDF30A9209DA333A0,773
NB 1050 DATA 00ADA03391CBC8A93D91CBC8A944
   91CBC8A200BDA33391CBE8C8C011F0034CF330
   EEA033ADA033C95BF0034C2031,219
EA 1060 DATA EEA033EEA033EEA033EEA033EEA0
   33EEA0332064334C4730CEA1332072 33A9008D
   A233A90B8D4203A5CB8D4403A5,851
DO 1070 DATA CC8D4503A9118D4803A9008D4903
   A2002056E4EEA233ADA233CDA133F047A90B8D
   4203A9B68D4403A9338D4503A9,710
BH 1080 DATA 018D48032056E4206433A90B8D42
   03A5CB8D4403A5CC8D4503A9128D4803A9008D
   49032056E4EEA233ADA233CDA1,593
SX 1090 DATA 33F0062064334C3131A9008DA133
   A99085CDA93785CEA210A9039D4203A9819D44
   03A9339D4503A9049D4A032056,16
WH 1100 DATA E4A210A9079D4203A9A09D4403A9
   339D4503A9019D4803A9009D49032056E4C99B
   F042EEA1332072 33A000B1CBC9,530
FT 1110 DATA 00F0D0CDA033D029B1CB99A333C8
   C013D0F6A202A000BDA33391CDE8C8E013D0F5
   18A5CD691085CDA5CE690085CE,594
GR 1120 DATA 4CC7312064334CF131A010A90091
   CD205233A90B8D4203A9838D4403A9338D4503
   A9188D4803A9008D4903A20020,735
XH 1130 DATA 56E4A99085D0A93785D1A9008D9E
   338D9F33A000B1D0D00160 18A5D069038D4403
   A5D169008D4503A9108D4803A9,834
RY 1140 DATA 008D4903A9098D4203A2002056E4
   205633A9039D4203A5D09D4403A5D19D4503A9
   049D4A032056E410034C3C33A2,979
OB 1150 DATA 20A9079D4203A9009D4403A9409D
   4503A9009D4803A9309D49032056E43007A901
   8D9E33D005A9008D9E33BD4803,592
TS 1160 DATA 8D9C338D49038D9D33AD9E33D003
   205633A001A93891D0AD9F33D01EA230A9039D
   4203A5D09D4403A5D19D4503A9,950
HS 1170 DATA 089D4A032056E4A9018D9F33A230
   A90B9D4203A9009D4403A9409D4503AD9C339D
   4803AD9D339D49032056E4AD9E,640
HG 1180 DATA 33F0034CAB322056332 05A3318A5
   D0691085D0A5D1690085D14C5A32A210D006A2
   20D002A230A90C9D42034C56E4,87
FE 1190 DATA 18A5CB691285CBA5CC690085CC60
   A90085CBA93485CC6044313A2A2E2A4B3A9B7D
   4D6F76696E6720746F2052616D,712
DH 1200 DATA 6469736B2E2E2E9B9B0000000000
   20000020202020202020202020202020202020
   202020200000E002E102003000,42
```

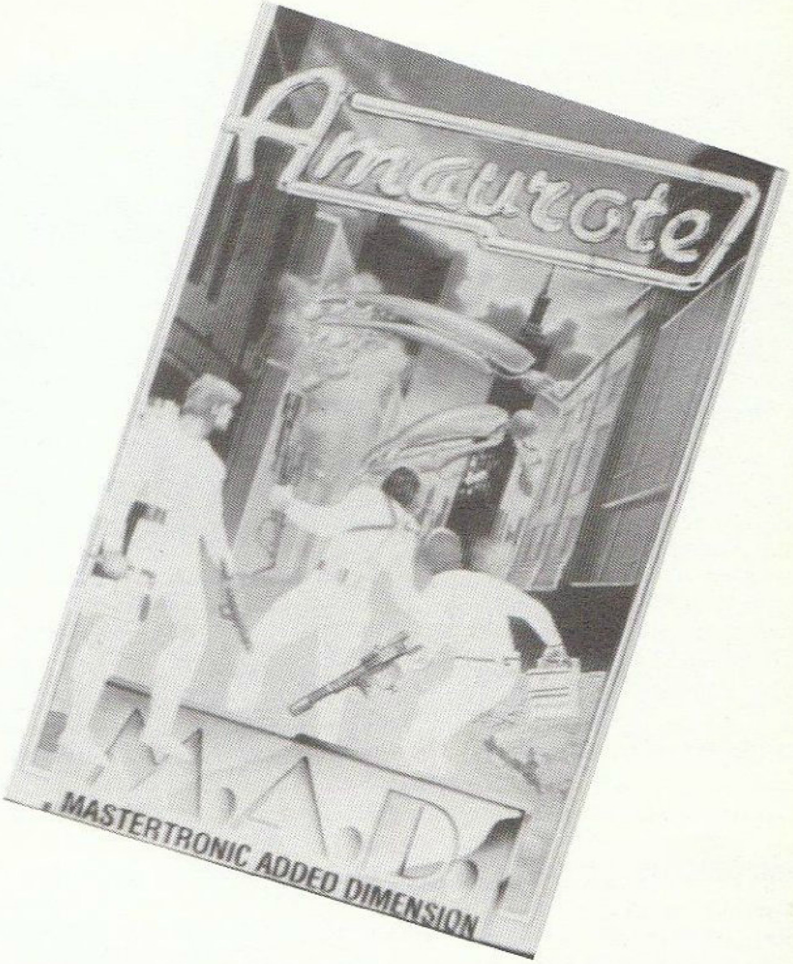
## Amaurote

From Mastertronic
Price £2.99 Cassette
Review by Gordon Cameron

This is another game from the same Mastertronic label that brought Atari users 'The Last V8' and 'Spellbound', so my expectations were understandably high. The cassette inlay spiel rambles on about how you, as an officer in the Royal Army of Amaurote, have the task of clearing all 25 districts of the city. Clearing them of what, I hear you ask. Well, supposedly the city has become infested by killer insects, and the population has taken flight. The army, after a bloody struggle, was forced to withdraw, and so it is up to you, as the last remaining uninjured officer, to take on the creatures single-handed.

So to the game itself, which incidentally takes an age to load. After the program has loaded, you are greeted by the credits, which are displayed on the viewscreen of your Arachnus 4 armoured car, accompanied by a stirring piece of music. The top 80% or so of the TV is taken up by the viewscreen, with the remainder taken by the control panel, with all your instruments. This is very detailed and quite difficult to make out at first. It consists of numerical readouts detailing your cash left, percentage damage to district, number of bombs remaining, percentage damage to the city as a whole, and damage to your craft. There are two additional instruments, the Scanner, which allows you to 'home in' on the different insects and bombs, and the Supabomb indicator, which flashes when you are carrying the Supabomb (funnily enough!). More of this later.

You start the game proper by selecting which of the 25 districts of the city you wish to visit first. The objective is to clear each of these sectors of all the insects, so it makes little difference which sector you start with. There are 3 different types of insect - Scouts fly around the city on the look out for food and intruders (i.e. you!), and then report back to the Queen your position. Drones are the most common and plentiful, and are sent out by the Queen to nab you. Once close to you, they are hard to shake off. Your craft is armed initially with 30 bouncing bombs which are effective against these insects. The last type of insect is the Queen herself. Conventional bombs don't work against her (typical!) so you have to radio an order for a Supabomb, which you then have to collect, homing in using the Scanner, and then fire at the Queen (whether it actually hits is another matter!). To order this so-called Supabomb, you press the OPTION key. This momentarily freezes the game action, and displays a menu of options. You then have the choice of ordering a Supabomb, replenishing your stock of ordinary bombs, repairing the Arachnus 4, or escaping from the current situation and being transported to a different position. All of these use up your cash. You start with 5 million. This may seem like a lot, but I can assure you it is depleted very rapidly!



After selecting the sector, you are greeted by a 3D view of the area, with your Arachnus 4 sitting in the middle. Your viewpoint is above and at a 45 degree angle to the action. The surroundings are detailed and have a very futuristic feel to them, with strange geometrical shapes abounding. Nudging your joystick moves your craft in the appropriate direction. Not only does it resemble a spider in looks, it also moves like one on its 4(!) pincer type legs. In the background, atmospheric chamber type music plays. This is impressive to start with, but grates after a while.

The animation is quite impressive and the view reminiscent of the Knight Lore/Alien 8 games of old. The screen doesn't scroll, but flips when you near the edge. Sooner or later, (more likely the former) you are likely to come across one of the insects, probably a Drone. These pad towards you and once spotted, you have a hell of a time trying to lose them. So you decide to let fly at them with one of your incredible bouncing bombs. You push the joystick towards the varmint and, with a smug smile, press the fire button, only to see the bomb soar out over the top of the insect and off the screen! You then have to wait until the bomb hits something (e.g. a building usually!) before you can fire again. By that time, the insect has probably caught up with you, and is rapidly draining your energy.

This is one of the major gripes I have. It is incredibly difficult to hit the thing you are aiming for, as the bomb takes an age moving, by which time the target has moved miles from the spot you were aiming for! Also, you have to move towards the target before you fire. This moves you closer to danger, and also means firing in certain directions is impossible in certain places, as you cannot move on to some pieces of scenery. An example - you are on a road between two pieces of impassable ground. Over one of these pieces of ground waltzes a Drone. What can you do? Answer, nothing as you cannot fire at it and it will undoubtedly catch up with you using a short-cut. Well, this is not strictly true, as you can escape by using the Rescue option on the radio screen. Nevertheless, it soon becomes tedious having to constantly call for rescue whenever you get into one of these zero options.

Complaint over(!), after playing for a while you can develop a technique to track down Drones using the Scanner, but it is by no means easy, usually requiring firing from one screen, blind onto another. After a time, you will no doubt be ready to take on the Queen herself. First you must order a Supabomb, which is dropped at a random location. (By the way, the incidental music which accompanies the radio screen and the zone select screen is excellent.) After locating and collecting this, you are ready to find and destroy the Queen. Note that the Supabomb is primed and armed when picked up, which means that you cannot use the normal bouncing bombs on route, so beware. Arrows on the scanner lead you to the Queen, which you must hit with your bomb. This can be tricky, as by this time she is probably onto your plan and has arranged a 'welcoming party' of Drones. It's a good idea to clear these before even picking up the Supabomb. Assuming you manage to destroy the Queen, wiping out the remaining insects in the sector is relatively easy, as they are no longer under her command. And then, all that remains is to do the same in all the other 24 districts. (And you've already used 1 million, tut, tut!).

Despite my criticisms, I enjoyed this game. The graphics are excellent, as is the animation, although it does slow down slightly with a few Drones on the screen. The presentation is superb, from the professional 'look' to the many extra touches, such as the Radio and the Zone selection screen, to the atmospheric music which accompanies the title & Radio screens, etc. The music during the game is good, but a bit repetitive. I found the movement a bit sluggish and quirky, making manoeuvres difficult at the best of times. The firing, as I've mentioned, is another sore point. It's too difficult to hit what you want to hit, as you have to be on exactly the same line, and the right distance away when you fire. Having said that, with practice, it IS possible to fool the Drones and escape them. The Repair and Rescue options are also very useful (and well used!). The game does play slowly, but then again, it is basically a strategy game with good graphics. It will take patience and a great deal of skill and time to complete, trying to keep city & district damage low, whilst keeping an eye on available funds! Despite my reservations, I kept on coming back for more (a glutton for punishment), which indicates good playability, which is perhaps the most important factor. In summary, great graphics, good sound, excellent music and presentation. A very professional product on the whole and well worth the £2.99 price tag it carries. At this price it's a steal, and I wholeheartedly recommend it. Keep up the good work, Mastertronic!

## Music Matrix

**Author: Lou Nisbet**
**Distribution: Music Matrix**
**Cost: £7.95 includes P & P**
**Computer: 800XL, 130XE and 1050 ONLY Review by Glissando**

It is not very easy to describe this program because it combines so many different features. It is a text adventure/educational game for music enthusiasts is probably the best I can come up with. Despite the use of a Disk Editor, to take the occasional crafty peek, I have not solved the riddle after many, many hours of use!

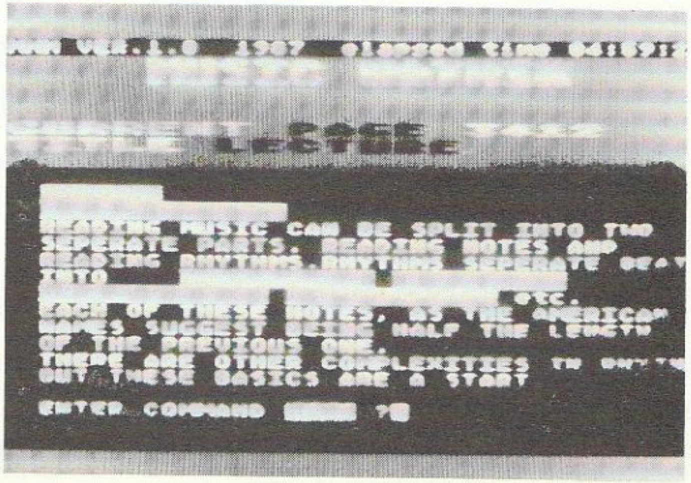
I will not spoil your enjoyment by giving any clues away, I had to search diligently for what success I have achieved and I see no reason to be THAT generous.

The command line recognises numbers, letters and words in order to pass through the matrix. Many screens require the use of Passwords to gain entry and these are the most difficult to find until you catch on to the modus operandi. But be warned, the reasoning behind the game is very subtle!

I mentioned in the description that it was educational too. For the student of music, you will definitely prove your worth when you have to call upon your knowledge to answer and solve some of the problems. In other parts of the game you will have to recognize and name chords from their SOUND. Incidentally, you have control over the difficulty of some of these riddles, making it a valuable learning tool for the newcomer right through to the experienced musician. As the advert says, it is an adventure, a course on playing and reading music, a random music generator and a tutorial on computer generated music.

In effect, the program consists of a number of small, interrelated programs introducing the various aspects of music and when each has been solved you will gain access to the structure, allowing you to create your own programs. The program runs with Turbo Basic, included on the disk.

This is a program that will give hours and hours of enjoyment. In adventure games it is quite common to allocate a number of months to the game's play ability, I doubt very much if it is possible to solve the game in six months, which will give you some idea of its value for money. If you are only just starting your journey into the world of music studying, this program will provide you with sufficient skills and problems to keep you occupied for a considerably longer period than this.



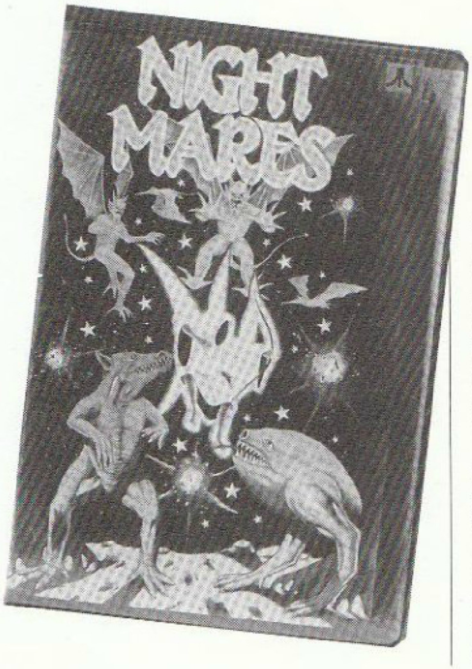
```
VER.1.0 1987
LECTURE

READING MUSIC CAN BE SPLIT INTO TWO
SEPERATE PARTS. READING NOTES AND
READING RHYTHMS.RHYTHMS SEPERATE
INTO
                                etc.
EACH OF THESE NOTES, AS THE AMERICAN
NAMES SUGGEST BEING HALF THE LENGTH
OF THE PREVIOUS ONE.
THERE ARE OTHER COMPLEXITIES
BUT THESE BASICS ARE A START

ENTER COMMAND       ?
```

## Nightmares

From Red Rat
Cassette £7.95 or disk £9.95
Review by D. Clapson

For some time now Red Rat have been producing 8 bit games by the dozen, but they often go a bit slack on the quality. Take Nightmares for example, the cover design leaves a lot to be desired for a start. And then there are the instructions, well lack of I should say.

The main idea of the game is similar to Dropzone, where you kill everything in sight. Although this is an unoriginal game concept, it is quite an addictive game. Mind you the smooth scrolling found in Dropzone is absent from Nightmares, in fact it often jolts and jiggers all over. The graphics, though reasonable, are just not up to Atari standard. This also applies to the sound.



The story goes like this: the lightness is going and evil creatures are appearing everywhere. Your land must be protected until the Lord Motohod (who thinks up these names?) returns. And like always you are the hero that is meant to save the world. With only a bow and arrow and lightning bombs you are supposed to kill off every type of evil that comes your way. Well I'm game!

You are also supposed to collect all the Arflits, which are a kind of super faery that live in swarms, as you go along.

The game control is via joystick and doesn't come up to standard. With all these bad points I reckon Red Rat will have a hard time making the top ten let alone the number one spot.

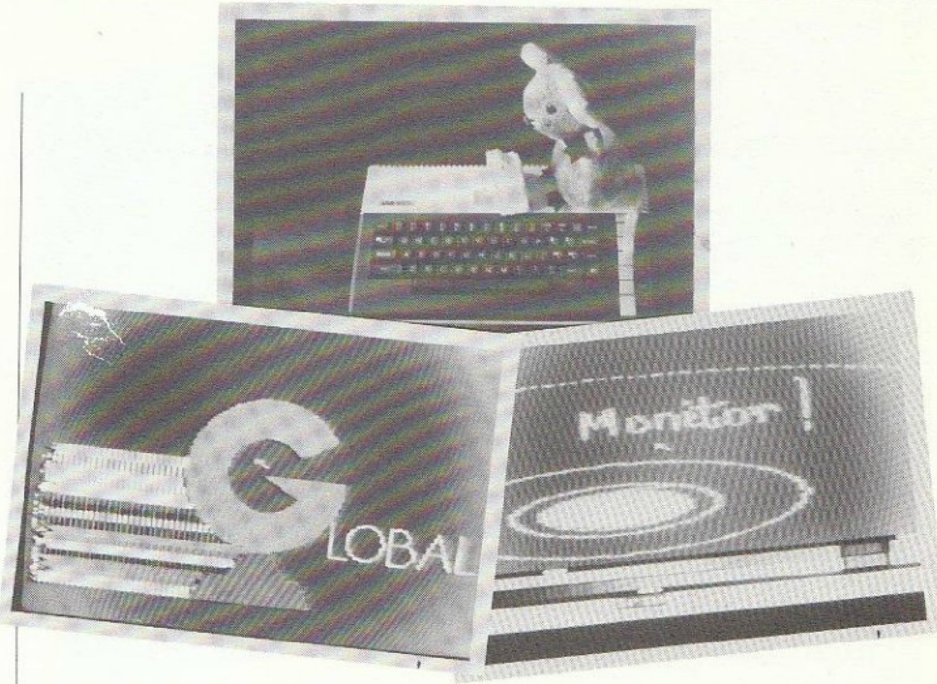
## XL/XE Mouse

Distribution: Global Computer Services, 108 Navigation Road, Northwich, Cheshire CW8 1BE.
Price: £29.95
Computer: XE/XL disk or cassette.
Review by Mike Stringer.

I do not think that a company has tried to produce a mouse option for this group of computers, until now. There have been numerous attachments available for some time to interface human actions to the computer. The device is a standard ST mouse which plugs into the number one port. There do not appear to be any alterations to the circuitry, etc. from a standard ST mouse, thus this product will be of considerable interest to ST owners as well. I have seen ST mice advertised for over £40 and when they breakdown, as any mechanical device will inevitably do, a convenient source of replacement is quickly sought - an optional extra for the XL/XE, but an absolute necessity for the ST!!

The product includes a D/S diskette, an ST mouse and a brief guide. Included amongst the programs on the disk is an Art Package. Upon booting up side A, with the mouse installed, the cursor flashes indicating that the mouse has been recognised and the READY stage appears. Using the guide, a variety of POKE options are detailed which allow you to familiarise with some of the features.

Loading the DEMO program, a small drawing program appears. With this you may draw with the left button and by clicking on the right button, you can page through a selection of colours from a small palette.



Also on this side, is a program called BIGDEMO, a somewhat expanded version of the first. It covers additional features. The guide explains clearly the various programming features, allowing modification and experimentation.

On the reverse is the ART PACKAGE with many menu features: FILL, CIRCLE, etc. It is also possible to load and save pictures to/from disk/cassette, there being quite a few already prepared. I feel that this product has appeared at the right time, no doubt some readers will be thinking it may be too late! Owners of the 8-bit series of computers are rightly depressed at the lack of enthusiasm for new programs/ideas for their machines. All the important software houses are directing their resources to the ST, almost to the point of totality!

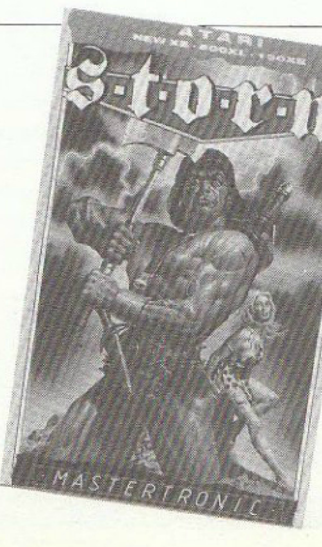
Throughout the guide, Global Computer Services encourage the user to experiment, and there is almost a plea from them for ideas for mouse environments, both commercially and non-commercially.

I hope that 8-bit users will give this product some thought. It works very well within the examples included, but the uses and potential is great. For the experimenter, the computerist who does not mind getting his feet wet, I think you will get a lot of pleasure from this product - it is excellent value and perhaps you may even get the opportunity to make a shilling or two on the way.

## Storm

From Mastertronic
Price £1.99 Cassette
Reviewed by D. Clapson

After waiting the usual 10 minutes for the tape to load I was confronted with the Rob Hubbard clone type of music, but no way as good as that on Jet Set Willy and Warhawk. But I've paid out £1.99 not just to hear pretty music, so how does the game rate? I'm placed in a maze type scenery, that of Gauntlet and the likes. One strange thing I noticed was that your character, and those of the enemy were very large compared to those other games. The idea of the game is to run about many a maze to collect three objects to get through a door to save your princess (haven't you heard that type of scenario before!).



Game play is via joystick and two players can work together, or if you wish to be anti-social then you can play on your own. I just couldn't get used to the controls, to change your mans direction you had to pull the joystick left for anti-clockwise and right for clockwise (although logical hard to master). Quite an easy game to complete but enough content to keep you happy for a few hours. Graphics are average and no sprites are used. Thumbs down on this one, I'm sad to say. You can do better Mastertronic.

# CLASSIFIED

The classified section is for private individuals only (not companies) to buy and sell computer hardware, software, make contacts, find pen pals, etc. All adverts will be free up to 30 words, thereafter the charge will be 10p per word (cheques and postal orders made payable to the club). Send your advert to us as soon as possible for the next issue together with any payment necessary. Please mark your envelope 'Classified'.

**Bargain!** Atari operating system Source Listing, as new still in wrapper, 400/800 version but suitable for 800XL owners. Only £10 including p&p. Ring (0702) 617307 after 6pm.

**8 Bit Original Software.** All disks. Astro Droid, Fighter Pilot, Spy vs Spy II, Silent Service, Solo Flight II. £3.50 each or £16.00 the lot. Ring (0274) 675739 after 6.30pm.

**Wanted.** Any service information (circuits, etc.) for 1050 disk drive. Contact Dave Austin, 26 Hastings Road, Ealing, London W13 8QH.

**Robotics.** Is anyone interested in robotics on the ST. I want to run up to 8 DC motors, can you help me? Write to W. Kingston, 120 Long Fellow Road, Worcester Park, Surrey KT4 8BD.

**For sale.** Atari 850 modem/printer interface unit including PSU, cables and instruction book, £50. 2 Bit Systems Midi Master disk and midi cable for 48K Atari. Phone Canterbury 67931 evenings.

**Pen Pals.** I wish to make contact, my name is Anthony Lewis, 57 Weir Terrace, Waiouru, New Zealand.

**Gateway Club.** We are a multi-user club that welcomes Atari owners. Membership is £7.00 per annum. For more details contact Phil Herberer, 164d Radcliffe Road, Lakenheath, Suffolk. Phone (Eriswell) 2363.

**Pen Pal.** I like a pen pal, roughly about 13 years of age, boy or girl. Write to R. Hall, 4 Highmead, Penllwyn, Pontlanfraith, Blackwood, Gwent, NP2 2PE.

**Wanted.** Tarot card program with the card graphics for the 130XE. Please contact R. Lussier, 7937 Elwell Street, Burnaby, B. C. Canada V5E 1M3.

**Please write!** My name is Jaroslaw Dolinski and I wish to correspond with other Atari owners (ST). My address is Polna 23, 83-110 TCZEW, Poland.

# ROUTINES MATTER

**By Marvey Mills Part 3**

In the last issue we started to take a look at how an assembly code programmer can access the disk drives on the ST. We used a very simple example, merely opening a file, reading it into the machine and then closing the file. In this article we will take a look at some more disk access routines and also the very important area of memory/buffer handling.

It is not good enough for us to be able to simply read in a disk file, we have to be able to read and write and we must also have the capability to create our own files and search for files already on the disk. Any program that uses disk input/output needs to set up some buffers somewhere in memory that can take all of the data read in or holds all the data ready to be written out. Last time we got round this problem by using the screen RAM as the buffer, which was okay because the data we were concerned with was a picture file. But what about in other cases? You have to allocate some memory yourself and then take care of it.

The subject of this issue's article is a simple little program that reads in a document file created using First Word, converts it to ASCII (more of ASCII later), and then writes it out to disk. Why do we need to do this? Well, I expect all of you have at one time or another clicked on a disk file with the DOC extension and been faced with the 'show','print' and 'cancel' dialog box. You will know that if you then click on the 'show' box the file will be printed to the screen for you to read. But if you attempt to do this to a First Word file it doesn't work properly. The words appear all bunched up together with no spaces between them, any centred text appears on the left-hand side of the screen uncentred and all the tabs have been lost, which makes it almost unreadable. You have to go to all the trouble of loading up First Word and then loading the file in just to read what may be a tiny document. This is because First Word saves the text in its own special way.

This article is the last in this three part series and completes our look at all of the basic functions a programmer needs to use in most simple programs. This one is a biggie, so sit down, take a deep breath and read on!

## More Disk Functions

If you take a look at the program listing you will recognise most of the introduction. We are still using the 'memory management header' that we discussed in the first part of the series. We have also added a new constant to the beginning called 'bell'. The value of this constant is 7 and if you print it to the screen using our print line routine it rings the bell. You will also see the familiar introductory code that disables the mouse, clears the screen and prints the program's title.

The first thing we do after this is get a line of input from the user using our getline routine from the last article. Here we are asking the user to supply us with the name of the file that we wish to convert. As in the past I have allowed only 20 characters to be entered in the getline routine. If the file you want to convert is in a folder then you may well want to enter more than this. If you have been following this series you should now understand how the getline routine works so I'm not going to tell you how to make this limit greater. Suffice to say that you will need to make an amendment to the source code in two different places. Go to it!

As in the last article, we use the fact that no characters were entered for the filename to signify that the user wants to quit the program. You should also know by now how we check whether this is the case.

The first unfamiliar piece of code is the section called 'search'. If we are going to read in a file from disk then we need to set up a buffer to hold it all as it is coming in. To do this properly we need to know the actual size of the file. We could of course just use all of the available memory but not only is this wasteful, it also doesn't leave any free memory for anything else and later on in the program we will need some more. So how do we get to know the size of a file on disk?

When a file is written to disk you will remember that other information is written with it, the File Attribute for example. We discussed the use of file attributes in the last article. Well the attribute is not the only thing that is saved. In order for the system to know all there is to know about the file, the date of its creation, the time of its creation, its size and its name are also saved. It would be silly if we had to read a file in just to read these bits of information and in this case it wouldn't work since we still don't know the size of the file, so GEMDOS provides us with a set of routines for interrogating this information. The first, called 'SFIRST', is used for setting up the search parameters and finding the first file on the disk that matches. The second, called 'SNEXT', will find all the others in turn that fit the parameters set up using 'SFIRST'. You may wonder why a search is needed at all since we already know which file we want, the user has just entered it! Well the good thing about both of these routines is that you don't need to exactly specify the filename. If you passed "*.*" as the filename to be searched for then the SFIRST routine would return the information on the first file on the disk. If you then called SNEXT repeatedly, it will return the information on all of the files on the disk in turn. You could use these routines in just this way to print the disk directory. However, since we do indeed know the filename we want we are only going to use SFIRST so that only the relevant information on that file is returned to us. By the way, if you want to try using an SFIRST/SNEXT combination in your own programs then you set up the search parameters using SFIRST only. You do not need to pass the search parameters to SNEXT, they are already set up from SFIRST.

When we call the SFIRST routine, quite a lot of information is returned to us. Where is GEMDOS going to put it all? Well we have to set up a 44 byte buffer known as the DTA or Disk Transfer Address and then when SFIRST has been executed we can then examine various parts of the DTA buffer for the information we require. See Listing 1 for a breakdown of the DTA contents. So the first thing we do is establish the address of the DTA buffer so that GEMDOS can use it. All we do is push the address of the 44 bytes we have reserved onto the stack, push the function number for SETDTA ($1A), and then call GEMDOS. After this function has been executed our DTA buffer is used in all subsequent SFIRST or SNEXT calls until we either change it ourselves or quit the program.

Once we have set up the DTA buffer we can set up the search parameters and call SFIRST to find our file. Push a word containing the attribute type of the file(s) to be searched for, and then push the address of the filename to be searched for. We have this filename in the input buffer from the getline routine so all we have to push is the address of lbuff plus 2 (to get over the buffer size bytes - see last issue) onto the stack and away we go. GEMDOS will search the disk for the filename and if it finds it, it will load the file information into the DTA buffer. If it can't find the file on the disk then it will return an error code in d0. Remember

| | |
|---|---|
| Bytes 0 - 20 | Reserved for GEMDOS use. |
| Byte 21 | The file attribute. |
| Bytes 22 - 23 | Time of file creation (taken from clock). |
| Bytes 24 - 25 | Date of file creation. |
| Bytes 26 - 29 | Size of file in bytes. |
| Bytes 30 - 43 | Name and extension of file. |

Listing 1. Format of DTA buffer.

that all GEMDOS error codes are minus values so all we do is test d0 for a minus value on return. If the file was found without problem then d0 contains zero on return.

## Creating Buffers

Once we have found the file and received the DTA information from GEMDOS we can easily find the length of the file. It is stored as longword in bytes 26-29 of the DTA buffer. Now we have to set aside an area of memory large enough to hold it all. We can't simply find a bit of free memory and bung it in there. Remember that GEM holds a list of reserved portions of memory and uses all the rest for itself so if we just wrote it straight into unused RAM then we run the risk of GEM overwriting it. What we have to do is ask GEM to reserve a chunk of memory for us and then leave it alone. The function to do this is called MALLOC (Memory ALLOCation). We can request any amount of memory from GEM and if there is enough free then GEM will add it to its internal list of reserved areas and pass the address of our buffer back to us in d0. Note that it ALWAYS sets the start of the buffer on an even address. If you ask GEM to reserve a greater amount of memory than is available, then MALLOC returns an error code in d0 (as usual it is a minus value but in this case it is a longword minus value). Another handy use of the MALLOC function is that if you pass -1 as the amount of memory you want to reserve, then GEM returns the amount of available RAM to you in d0 rather than reserving some memory.

So to allocate memory for our file, all we do is get the length of the file out of the DTA buffer and pass it to MALLOC. If it can, then MALLOC reserves this amount of space and tells us where in memory our buffer starts. In fact in our program we have to do this twice, once for the buffer to read our text in, and then again to allocate space in which we will build up the converted file before writing it out to disk. We use the same size for both of the buffers.

Once we have performed the conversion and written the file out to disk, then we are going to re-run the program so that the user can convert another file if he so wishes. This in turn will create a new set of input and output buffers leaving the old ones still in memory wasting space. If we were to carry on doing this we would eventually run out of RAM. The thing is that GEM reserves the buffers until we specifically allow it to release the memory back into its 'pool' OR we quit the program. If we quit then GEM automatically de-allocates any memory we have grabbed within the program. Since we will not necessarily be quitting we need a way of telling GEM that we no longer need the RAM we have reserved. The function to do this is called MFREE and its function number is $49. All you do is pass GEMDOS the address of the start of the buffer and it will clear that entry from its table of reserved blocks of RAM. See Listing 2 for a complete listing of buffer handling using MALLOC and MFREE.

```
;* Check amount of free RAM

        move.l  #-1,-(sp)       ;inquire amount of free ram
        move.w  #$48,-(sp)      ;push MALLOC function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack.
                                ;d0 now contains the amount of free
                                ;RAM in bytes.

;* Reserve 1024 bytes for a buffer

        move.l  #1024,-(sp)     ;push number of bytes required
        move.w  #$48,-(sp)      ;push MALLOC function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack
        tst.l   d0              ;was it able to do so?
        bmi     error           ;minus value so not able to reserve
        move.l  d0,buffstrt     ;else d0 contains address of buffer
                                ;start.

;* Release buffer back to GEM

        move.l  buffstrt,-(sp) ;push address of start of buffer
                                ;that you want to release
        move.w  #$49,-(sp)      ;push MFREE function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack
        tst.l   d0              ;error releasing block?
        bne     error           ;d0<>0 so error occured
```

Listing 2. Buffer allocation usage.

## Even More Disk Functions

We already know the procedure for opening, reading and closing a file. What about when we want to write information to a file as well? When we call the OPEN function what we are really saying to GEMDOS is pick a file that exists already as a directory entry and allow use to perform an operation on that file. In the last article we opened an existing picture file on the disk and then proceeded to use READ for getting the data off the disk and into memory. When we have converted our First Word file into ASCII we will want to be able to write it back to the disk again to be used later. We will know the filename since we have prompted the user to supply it. But if the given name is a brand new one and does not already exist on the disk then GEMDOS doesn't know anything about it. How can we tell GEMDOS to open a file that doesn't exist yet! What we have to do is create a new file in the disk directory and, unsurprisingly, the function to do this is called CREATE, its function number is $3C. All you have to do to create your brand new file is push the, now familiar, attribute word onto the stack followed by the address of the filename of the file you wish to create. Call GEMDOS and all being well the handle of the new file will be returned to you in d0. If there were any problems in creating the file then you will get an error code in d0 instead. A note of caution here, if you attempt to create a file that already exists on the disk then the old one will be unrecoverably erased and the name re-used for the new file. So beware when using this function.

When you create a new file, GEMDOS assumes you have done this because you actually want to use it as well! It leaves the file open for your use once it has created it so you do not then have to go and OPEN it before you use it. Remember the handle has been passed to you via the d0 register so you can get right on with the job of writing the data. WRITE works in precisely the same way as the READ function only in reverse. You supply the address of the buffer from which the write is to take place, followed by the amount of bytes to write and the handle of the file that you want to write the data to. Following the write call d0 will either contain a minus value (longword) error code if there was a problem (disk full for example) or the number of bytes successfully written to the file. The function number for WRITE is $40, see Listing 3 for the general usage of the CREATE and WRITE functions. Remember that as with the READ call, all files have to be closed once you have finished with them.

```
;* Create a file called 'TEST.DAT' on the A: drive.

        move.w   #attribute,-(sp)    ;push desired attribute value
        move.l   #filename,-(sp)     ;push address of filename
        move.w   #$3c,-(sp)          ;CREATE function number
        trap     #1                  ;call GEMDOS
        add.l    #8,sp               ;correct stack
        tst.w    d0                  ;did an error occur?
        bmi      error               ;if d0 is minus the error
        move.w   d0,handle           ;else d0 contains file handle
        .
        .
filename  dc.b 'A:TEST.DAT',0        ;filename. Delimited by a zero
                                     ;byte.

;* Write 1024 bytes to a file.

        move.l   #buffer,-(sp)       ;address of buffer to write
                                     ;from
        move.l   #1024,-(sp)         ;1024 bytes to write
        move.w   handle,-(sp)        ;handle of file to write to
        move.w   #$40,-(sp)          ;WRITE function number
        trap     #1                  ;call GEMDOS
        add.l    #12,sp              ;correct stack
        tst.l    d0                  ;did an error occur?
        bmi      error               ;if d0 is minus then error
                                     ;else d0 contains number of
                                     ;byte written to file
```

Listing 3. CREATE and WRITE usage.

## When is a Space Not a Space?

Right then, we have found out how we can reserve buffers exactly big enough to take a disk file and how to get more information about a disk file. We've found out how to create new files and write information to them once we have done so. What we have still left to learn is how we actually convert the data from First Word to ASCII format. And what is ASCII anyway?

ASCII as you may know, stands for American Standard Codes for Information Interchange, now you know why its shortened to ASCII! All it is, is a standard set down describing which values should be used to denote which characters. For example, the character 'W' has the value 87 (decimal) as its ASCII code while the character 'w' has 119 (decimal) as its ASCII code. All of the characters of the alphabet, both upper and lowercase, plus all of the numbers from zero to nine each have their own code. So too do all of the other characters such as ( and " and $. The value we use to make the bell ring is also an ASCII character, as I said above the actual value is 7 (decimal). So as well as all of the characters of the alphabet, some of the codes are said to be non-printable, in other words they are used by the computer to denote something other than alphanumeric characters. The ESC code (27) is a good example of a non-printable character.

Our specific problem arises out of the fact that First Word can do fancy things like automatic right justification of text or keep a title centred automatically. When you type a normal space into First Word what it actually puts into the text is not the value for a space character (32 decimal) but another value that the program itself knows is supposed to be a space character. Similarly, when you centre a piece of text the program puts yet another code in the text instead of a space to denote that the spaces pushing the text into the middle of the page are subject to later changes in number. It does this so that, prior to printing, it can work out how many real spaces are needed to be printed instead of just the one that we have typed in. The values it uses for these functions are $1E for a 'soft' space and $1C for a 'centred' space. Both of these codes are valid ASCII codes but they are non-printable so when you attempt to print a raw First Word file to the screen, the screen ignores them since it has no printable character equivalent. All we have to do then is to look through the file exchanging both of the above values with a valid space character.

Well actually it isn't quite as simple as that, as well as the codes discussed it also puts ESCape codes into the text which control the printer functions. When these codes are sent to the printer, it first sees the ESC value and knows that the next character to follow is actually a command to make it do something different, like go into bold text for example. So we also have to fish these out as well. The only other problem is that when First Word saves a file to disk it also saves a load of information relating to the document itself at the front of the file. We have to skip over this to get to the text itself. First Word always saves an image of the ruler used when creating the text and the last character in the ruler image is always a 'close square bracket' character. Straight after this is one byte of data containing information about the font to be used when printing the document. So all we have to do is look for the close square bracket symbol, skip one more byte, et voila! We are at the start of the text. This also provides us with a handy way of making sure that the file is in fact a true First Word file. If we have looked all the way through the document and we haven't found the close square bracket character then we know that the file is not in the correct format for converting.

So finally, the steps we take in this program (Listing 4) are as follows. We get the name of a file to be converted from the user. We check its size, set up an input buffer to hold it all, read it in and close the file. We also set up an output buffer of the same size (the document can NEVER be longer after conversion). Then we look through the buffer byte by byte until we find a close square bracket character and then skip one more byte. We are now at the text itself. We get each byte from the input buffer and if it is not a 'soft' space, a 'centred' space or an ESCape value we copy it to the output buffer. If it is one of the above we just write a space to the output buffer or skip the ESC code and following value if it was an ESCape code. Every time we write a character to the output buffer we increment a counter so that we know how many bytes to write out afterwards. When all of the characters in the input buffer have been checked and copied or discarded we tell the user that the conversion is complete and ask for a filename to write the new data to. Then we create this file on the disk, write the output buffer to it and finish up by closing the file. At this point we give all the buffer space used back to GEM and loop round to the start in order to do another document.

I'm sorry if this has got a bit complicated. If all you want to do is learn buffer control and disk input/output then just ignore the explanation of the structure of First Word files and play around with the routines discussed. If you have been following this series then you should now have all you need to know to get you going writing programs of your own. GEM contains many, many more very useful routines which I have not attempted to cover here. There are, for example, another eleven routines in GEMDOS concerned with disk I/O and file handling alone! If this series has whet your appetite for assembly language programming on the ST then there are a couple of books that I would suggest you seriously consider buying for further information. The first is 'The Concise Atari ST 68000 Programmer's Reference Guide' published by Glentop. The second is 'Anatomy of the Atari ST' published by 1st Publishing (this book is also known as 'Atari ST Internals' published by Abacus Software). I wish you luck.

Listing 4.

```
;* TXTCON - By Marvey Mills for Monitor Magazine
;* A First Word to ASCII text converter

lf      equ     $0a             ;set up some constants
cr      equ     $0d             ;for the print formatting
eol     equ     $0              ;characters - makes program
esc     equ     27              ;more readable
bell    equ     7
nomouse equ     $a00a           ;A-Line call - hide mouse cursor

;** Memory Management Header **

        move.l  a7,a5           ;get old stack pointer
        move.l  #ourstack,a7    ;and set stack pointer to our area

        move.l  4(a5),a5        ;add up all the various parts
        move.l  $c(a5),d0       ;of memory that our program
        add.l   $14(a5),d0      ;uses and tell GEM to keep them
        add.l   $1c(a5),d0      ;safe for our program
        add.l   #$100,d0        ;otherwise GEM could overwrite the
        move.l  d0,-(sp)        ;program during its
        move.l  a5,-(sp)        ;operation
        move.w  #0,-(sp)
        move.w  #$4a,-(sp)
        trap    #1
        add.l   #12,sp

;******************************
;     PROGRAM CODE HERE       *
;******************************

start
        dc.w    nomouse         ;hide the mouse
        jsr     clrscrn         ;then clear the screen

        move.l  #title,a0       ;get address of title string
        jsr     pline           ;and call our print line routine
getin
        move.l  #input,a0       ;get address of input name prompt
        jsr     pline           ;and call out print line routine

        jsr     getline         ;get the input filename
        cmp.b   #0,lbuff+1      ;charcters entered?
        beq     quit            ;no - so quit program

search
        move.l  #dta,-(sp)      ;get address of DTA buffer
        move.w  #$1a,-(sp)      ;SETDTA function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack pointer
```

```
        move.w  #0,-(sp)        ;files with #0 attribute to be found
        move.l  #lbuff+2,-(sp)  ;address of filename in lbuff
        move.w  #$4e,-(sp)      ;SFIRST function number
        trap    #1              ;call GEMDOS
        add.l   #8,sp           ;correct stack
        tst.w   d0              ;error?
        bmi     notfnd          ;yes- file not found.

getlen
        move.l  dta+26,d0       ;get file length from DTA buffer
        move.l  d0,fsize        ;save length for later

resin
        move.l  d0,-(sp)        ;reserve a block the size of the file
        move.w  #$48,-(sp)      ;MALLOC function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack
        tst.l   d0              ;error?
        bmi     memerr          ;yes- unable to reserve space
        move.l  d0,inbuff       ;no- save address of input buffer

resout
        move.l  fsize,-(sp)     ;reserve output buffer of same size
        move.w  #$48,-(sp)      ;MALLOC function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack
        tst.l   d0              ;error?
        bmi     memerr          ;yes- unable to reserve space
        move.l  d0,outbuff      ;no- save address of output buffer

open
        move.w  #$0,-(sp)       ;set file attribute to read/write
        move.l  #lbuff+2,-(sp)  ;address of filename in lbuff
        move.w  #$3d,-(sp)      ;OPEN function number
        trap    #1              ;call GEMDOS
        add.l   #8,sp           ;correct stack
        tst.w   d0              ;error?
        bmi     dskerr          ;yes- inform user
        move.w  d0,handle       ;no- save file handle
read
        move.l  inbuff,-(sp)    ;get address of buffer for read
        move.l  fsize,-(sp)     ;get number of bytes to read
        move.w  handle,-(sp)    ;get file handle
        move.w  #$3f,-(sp)      ;READ function number
        trap    #1              ;call GEMDOS
        add.l   #12,sp          ;correct stack
        tst.l   d0              ;error?
        bmi     dskerr          ;yes- inform user
close2
        move.w  handle,-(sp)    ;get file handle to close
        move.w  #$3e,-(sp)      ;CLOSE function number
        trap    #1              ;call GEMDOS
        add.l   #4,sp           ;correct stack
        tst.w   d0              ;error?
        bmi     dskerr          ;yes- inform user.

;*************************************************
;* file has now been read into the input buffer
;* and the output file has been set up to take
;* the converted output.
;* Now we perform the actual conversion on the
;* text, writing it to the output buffer as we go.
;*************************************************

        move.l  inbuff,a0       ;get input buffer address to a0
        move.l  outbuff,a1      ;get output buffer address in a1
        move.l  #0,count        ;clear output buye count
        move.l  fsize,d2        ;get number of bytes to process

rloop
        move.b  (a0)+,d1        ;get a byte from input buffer
        cmp.b   #']',d1         ;is it the end-of-ruler character?
        beq     endrule         ;yes- go and start conversion
        sub.l   #1,d2           ;no- correct no. bytes left to do
        cmp.w   #0,d2           ;have we run out of bytes?
        bne     rloop           ;no- get the next one
        bra     notfw           ;yes- file not First Word

endrule
        sub.l   #1,d2           ;correct no. bytes left to do
        move.b  (a0)+,d1        ;discard the 1 byte font information
        sub.l   #1,d2           ;correct no. bytes left to do

cloop
        move.b  (a0)+,d1        ;get a byte of text from buffer
        cmp.b   #$1e,d1         ;is it a 'soft' space?
        beq     sspac           ;yes- process it
        cmp.b   #$1c,d1         ;is it a centering space?
        beq     sspac           ;yes- process it
        cmp.b   #$1b,d1         ;is it an escape code?
        beq     procesc         ;yes- process it

wrtasc
        move.b  d1,(a1)+        ;write the byte in d1 to output buff
        add.l   #1,count        ;increment output character count
        sub.l   #1,d2           ;correct no. of bytes left to do
        tst.l   d2              ;are we done yet?
        bmi     convdun         ;yes- go and write file
        bra     cloop           ;else go and get next character

;*********************************************
;* the file has now been converted to ASCII
;* format and is in the output buffer.
;* Now we write the file out to disk.
;*********************************************
convdun
        move.l  #dunit,a0       ;inform user of completion
        jsr     pline
getout
        move.l  #output,a0      ;get address of output file prompt
        jsr     pline           ;and print it
        jsr     getline         ;get output filename from user
        cmp.b   #0,lbuff+1      ;name entered ok?
        beq.s   getout          ;no- print prompt again
```

```
create
        move.w  #0,-(sp)        ;file wil have read/write attribute
        move.l  #lbuff+2,-(sp)  ;get address of filename from lbuff
        move.w  #$3c,-(sp)      ;CREATE function number
        trap    #1              ;call GEMDOS
        add.l   #8,sp           ;correct stack
        tst.w   d0              ;error?
        bmi     dskerr          ;yes- inform user
        move.w  d0,handle       ;save handle of new file

write
        move.l  outbuff,-(sp)   ;get address of output buffer
        move.l  count,-(sp)     ;get number of bytes to write
        move.w  handle,-(sp)    ;get file handle to write to
        move.w  #$40,-(sp)      ;WRITE function number
        trap    #1              ;call GEMDOS
        add.l   #12,sp          ;correct stack
        tst.l   d0              ;error?
        bmi     dskerr          ;yes- inform user

close
        move.w  handle,-(sp)    ;get file handle to close
        move.w  #$3e,-(sp)      ;CLOSE function number
        trap    #1              ;call GEMDOS
        add.l   #4,sp           ;correct stack
        tst.w   d0              ;error?
        bmi     dskerr          ;yes- inform user


endit
        move.l  #done,a0        ;get address of ending text to a0
        jsr     pline           ;print it
        jsr     getkey          ;get a key
        jsr     givebuf         ;give the buffers back to GEM
        jmp     start           ;and re-run the program


;*******************************************
;* Specialized routines for this program   *
;*******************************************


notfnd
        move.l  #nofile,a0      ;get address of not found message
        jsr     pline           ;print it
        jmp     getin           ;go and get another filename

notfw
        jsr     givebuf         ;release buffers to GEM
        move.l  #wrong,a0       ;get address of error message
        jsr     pline           ;print it
        jsr     getkey          ;get a keypress
        jmp     start           ;and re-run program

dskerr
        jsr     givebuf         ;release buffers to GEM
        move.l  #errmsg,a0      ;get address of error message
        jsr     pline           ;print it
        jsr     getkey          ;get a keypress
        jmp     start           ;and re-run program

memerr
        move.l  #memmsg,a0      ;get address of memory error message
        jsr     pline           ;print it
        jsr     getkey          ;get a keypress
        jmp     quit            ;and quit program

sspac
        move.b  #' ',d1         ;substitute an ASCII space in d1
        bra     wrtasc          ;and go to write it out

procesc
        add.l   #1,a0           ;bump input pointer over ESC code
        sub.l   #2,d2           ;correct no. of bytes left to do
        bra     cloop           ;do not write d1, get next char

givebuf
        move.l  inbuff,-(sp)    ;get address of input buffer
        move.w  #$49,-(sp)      ;MFREE function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack

        move.l  outbuff,-(sp)   ;repeat function for output buffer
        move.w  #$49,-(sp)
        trap    #1
        add.l   #6,sp

        rts                     ;return to program


;**********************************************************************
;* START OF SUBROUTINE LIBRARY                                        *
;**********************************************************************


clrscrn
        move.l  #scrncode,a0    ;get address of ESC code for clear
        jsr     pline           ;call our print line routine
        rts                     ;return to program
scrncode
        dc.b    esc,'E',eol     ;escape code for clear screen
        even

quit
        move.w  #0,-(sp)        ;push function number on stack
        trap    #1              ;call GEMDOS

pline
        move.l  a0,-(sp)        ;get address to start print
        move.w  #$09,-(sp)      ;push the function number on stack
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack
        rts                     ;return to program

getkey
        move.w  #$01,-(sp)      ;push function number on stack
        trap    #1              ;call GEMDOS
        add.l   #2,sp           ;correct stack

        and.b   #%11011111,d0   ;this line to forces uppercase

        rts                     ;return to program with character

getline
        move.l  #lbuff,a0       ;get address of line buffer
        move.b  #20,(a0)        ;initialize max size
```

```
        move.l  a0,-(sp)        ;and push address on stack
        move.w  #$0a,-(sp)      ;push readline function number
        trap    #1              ;call GEMDOS
        add.l   #6,sp           ;correct stack
        rts                     ;and return to program

;*************************
;* Text storage
;*************************

title
        dc.b    '**************************************',cr,lf
        dc.b    '*                TXTCON              *',cr,lf
        dc.b    '*                                    *',cr,lf
        dc.b    '*            By Marvey Mills.        *',cr,lf
        dc.b    '*   For Monitor Magazine. 08/11/87   *',cr,lf
        dc.b    '*  A First Word to ASCII converter.  *',cr,lf
        dc.b    '**************************************',cr,lf,eol

input   dc.b    cr,lf,lf
        dc.b    'Enter filename for conversion.',cr,lf
        dc.b    '>',eol
output  dc.b    cr,lf
        dc.b    'Enter filename for converted file.',cr,lf
        dc.b    '>',eol
errmsg  dc.b    cr,lf,lf
        dc.b    'A disk error has occurred.',cr,lf,lf
        dc.b    'Press any key to re-run...',eol
memmsg  dc.b    cr,lf,lf
        dc.b    'Unable to allocate sufficient memory.',cr,lf
        dc.b    'Press a key to quit program...',eol
dunit   dc.b    cr,lf,lf
        dc.b    bell,'Conversion complete!',cr,lf,eol
nofile  dc.b    cr,lf,lf
        dc.b    bell,'Error- File not found!',cr,lf,eol
wrong   dc.b    cr,lf,lf
        dc.b    bell,'Error- File not First Word format.',cr,lf
        dc.b    'Press a key to rerun...',cr,lf,eol
done    dc.b    cr,lf,lf
        dc.b    bell,'File converted and written ok.',cr,lf
        dc.b    'Press a key to rerun...',cr,lf,eol

;*************************
;* data storage
;*************************
                even
                ds.l    32      ;user stack area
ourstack        ds.l    1
handle          ds.w    1       ;save file handle
lbuff           ds.b    22      ;line input buffer
inbuff          ds.l    1       ;pointer to buffer for input.
outbuff         ds.l    1       ;pointer to buffer for output.
dta             ds.b    44      ;disk transfer address buffer
fsize           ds.l    1       ;size of input file
count           ds.l    1       ;output character counter


        end
```

## Enduro Racer

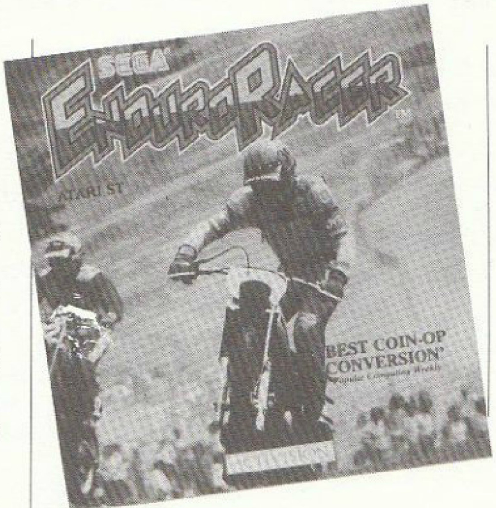
**From Activision**
**Price £19.99**
**Review by Harry Prince**

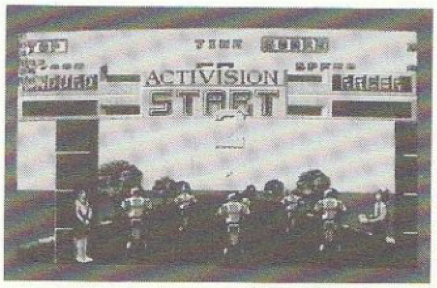
At last the ST version of Enduro Racer is out, it's been available on other systems for a while now and at one point I thought we were never going to see it. Well is it worth the wait? The answer is a firm YES!! The graphics are great, the action fast and furious, and the conversion from coin-op classic has been faithful to the original.

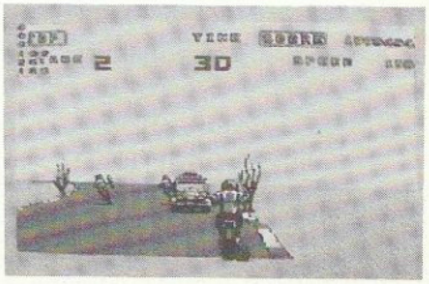
In Enduro Racer you take the part of a motorbike racer and you must compete against other riders and complete each course in under 60 seconds, at the same time avoiding obstacles such as piles of rocks in the road, jumping over humps and landing safely, missing the trees and hoardings that line the side of the road, and it would be nice to stay on the road too! If you manage to finish the course in time you move on to a bonus track for even more high speed action.

Control of the bike is by joystick or mouse, moving left and right causes your man to turn in that direction with his foot on the ground for added balance, just like the real thing. Forward on the stick opens the throttle to increase your speed, the fire button (left button on mouse) are the brakes, pulling back causes a wheelie (a must when jumping the humps). Don't brake on landing though, your man will fly over the handlebars! There are 5 stages to get through sporting different landscapes, not an easy task by any means.

Enduro Racer is a quality game, that won't pall after 5 minutes, has good graphics, immense playability and is a worthy addition to any software collection. Not just for bikers, fun for everybody!







## Super Sprint

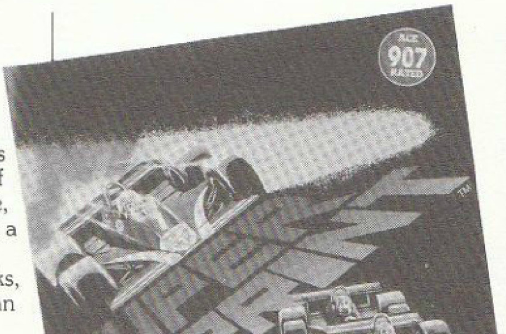
From Electric Dreams
Price £14.99
Review by Colin Thoms

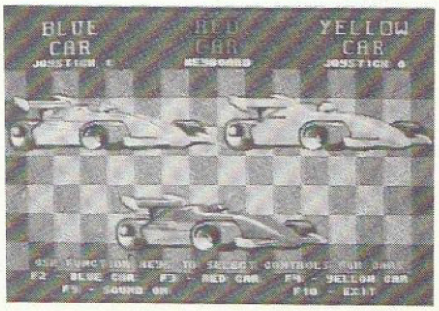
This transport from the Atari Games arcade coin-op is a stunning example of how ported over games should be done, it is sleek, very close to the original and a joy to play. Up to 3 players compete against each other over 8 different tracks, each one with more twists and turns than the last, and there are 4 levels of difficulty. Each track has an assortment of hazards including oil slicks, water spots, poppers, ramps to jump, opening and closing gates, hidden short cuts and strong gusty crosswinds (well it's a whirlwind actually).

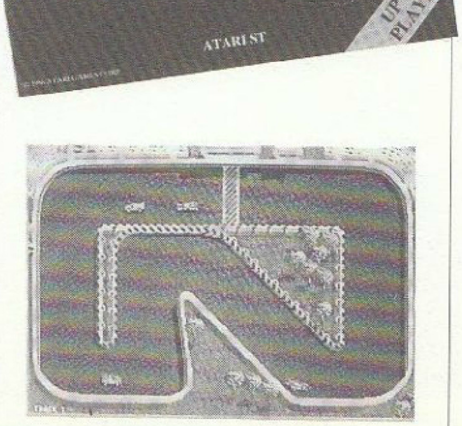
There are 4 racing cars in each race, coloured blue, red, yellow and grey. Two can be driven via joysticks, a third by keyboard operation, and the fourth is controlled by the computer. If you play on your own the computer takes over control of the others. Left and right on the stick give your car direction and the fire button gives the speed (beware there are NO brakes). The minimum lap count is 4, but some tracks are longer. The race is started by flag and you're away.

At random places and times during the race Bonus Score Pads appear, run over one and your score is increased. Also making an appearence are Golden Spanners which you collect in the same way, if you get three in one race you can customise your car in time for the next race, you can add super traction, higher top speed or turbo acceleration, but what ever you do, just win! Winning is the key, the more you win the more you get! During the race you will probably hit the circuit barriers many times, doing this at relatively low speeds only delays you somewhat, throughing up little dust clouds of desparation. Hitting a barrier at high speed however completely obliterates your car, luckily you are unharmed and you must wait for a new car to be delivered to you by helicopter, then you can continue with the race.

My conclusion must be that Super Sprint is just that: SUPER!







## DXpert (V1.4)

Author: Laurence Wilkes
Publisher: Five Pin Din, 37 Acacia
Avenue, Owlsmoor, Camberley,
Surrey GU15 4YH.
Price: £32.75
ST 520/1040 MONO, High
Resolution ONLY.
Review by Glissando

The DXpert is a voice editor for the Yamaha DXTX 7,S,D,FD synthesisers and expanders, and I believe it may also be of some use to FB01 owners, although I have not had the opportunity to try one out yet!

For many owners, the DX7 is a very difficult synthesiser to program. It is common knowledge that many owners would prefer to purchase a second model rather than face the task of facing FM synthesis, or to purchase the little RAM packs at £90 a throw!!

I have been the proud owner of one of these formidable beasts now for about four years and it took me about six months to come to terms with its sound structure. On average, it would take about four to six days to create a new sound. One could quickly modify an existing voice simply by changing output levels and algorithms, but this is not creation - merely modification. There are a few programs available to assist the voice programmer, but these are notoriously expensive - almost to the point of being ludicrous.

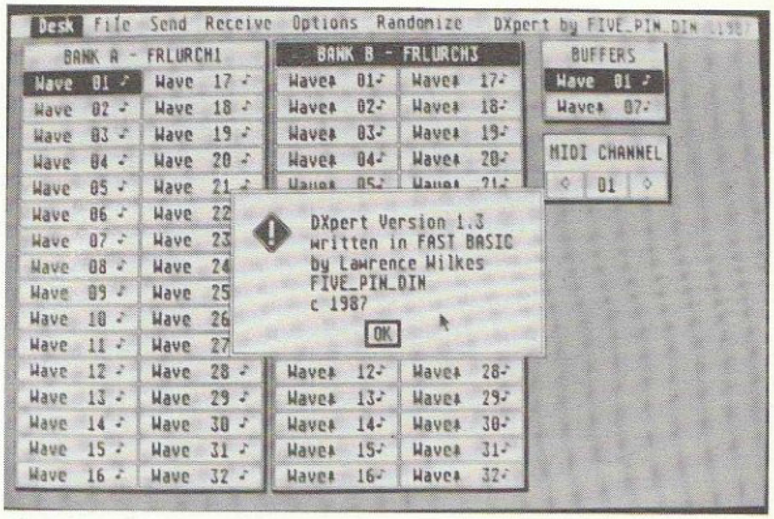
DXpert has reduced voice production from one of patient tedium, to one of pleasure.

On booting the .PRG, the desk shows two banks of thirty-two voice slots, a buffer for eight voices and a MIDI Channel selector defaulted to MIDI Channel 1. The menu selection includes FILE, SEND, RECEIVE, OPTIONS and RANDOMIZE.

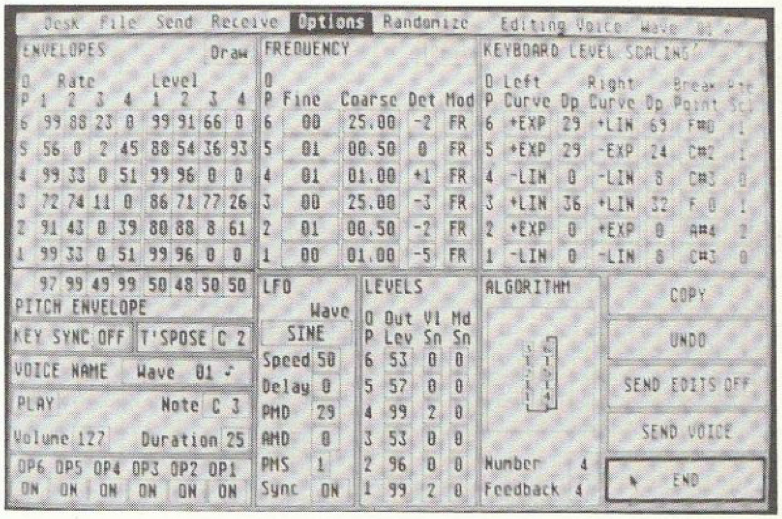
FILE is a standard LOAD, SAVE file device plus a neat toggle that allows you to pre-select the destination drive, A or B. This is the first time I have come across this useful switch on a menu.

Because of the various, essential, differences between the synthesiser and expanders, two methods of sending data from the ST to the external devices are to be found - one for the synthesiser and one for the expander. Similarly, the same argument holds true for RECEIVE.

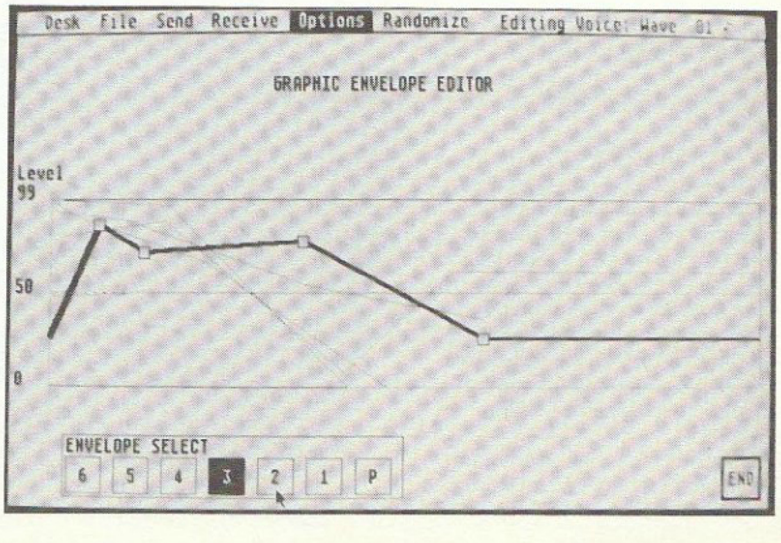
Under the OPTION menu are to be found EDIT VOICE, PRINT ACTIVE VOICE, PRINT ACTIVE BANK and HELP.



```
BANK A - FRLURCH1        BANK B - FRLURCH3        BUFFERS
Wave 01 ♩  Wave 17 ♩     Wave♦ 01♩  Wave♦ 17♩     Wave  01 ♩
Wave 02 ♩  Wave 18 ♩     Wave♦ 02♩  Wave♦ 18♩     Wave♦ 07♩
Wave 03 ♩  Wave 19 ♩     Wave♦ 03♩  Wave♦ 19♩
Wave 04 ♩  Wave 20 ♩     Wave♦ 04♩  Wave♦ 20♩     MIDI CHANNEL
Wave 05 ♩  Wave 21 ♩     Wave♦ 05♩  Wave♦ 21♩     ◇  01  ◇
Wave 06 ♩  Wave 22
Wave 07 ♩  Wave 23
Wave 08 ♩  Wave 24
Wave 09 ♩  Wave 25
Wave 10 ♩  Wave 26
Wave 11 ♩  Wave 27
Wave 12 ♩  Wave 28 ♩     Wave♦ 12♩  Wave♦ 28♩
Wave 13 ♩  Wave 29 ♩     Wave♦ 13♩  Wave♦ 29♩
Wave 14 ♩  Wave 30 ♩     Wave♦ 14♩  Wave♦ 30♩
Wave 15 ♩  Wave 31 ♩     Wave♦ 15♩  Wave♦ 31♩
Wave 16 ♩  Wave 32 ♩     Wave♦ 16♩  Wave♦ 32♩

DXpert Version 1.3
written in FAST BASIC
by Lawrence Wilkes
FIVE_PIN_DIN
c 1987
[OK]
```



```
ENVELOPES                 Draw  FREQUENCY                      KEYBOARD LEVEL SCALING
O  Rate         Level          O                              O Left        Right      Break  Rte
P  1  2  3  4   1  2  3  4     P Fine  Coarse  Det  Mod       P Curve  Dp  Curve  Dp  Point  Scl
6 99 88 23  0  99 91 66  0     6  00    25.00   -2   FR       6 +EXP   29  +LIN   69  F#0    1
5 56  0  2 45  88 54 36 93     5  01    00.50    0   FR       5 +EXP   29  -EXP   24  C#2    1
4 99 33  0 51  99 96  0  0     4  01    01.00   +1   FR       4 -LIN    0  -LIN    8  C#3    0
3 72 74 11  0  86 71 77 26     3  00    25.00   -3   FR       3 +LIN   36  +LIN   32  F 0    1
2 91 43  0 39  80 88  8 61     2  01    00.50   -2   FR       2 +EXP    0  +EXP    0  A#4    2
1 99 33  0 51  99 96  0  0     1  00    01.00   -5   FR       1 -LIN    0  -LIN    8  C#3    0
  97 99 49 99  50 48 50 50     LFO         LEVELS             ALGORITHM            COPY
PITCH ENVELOPE                        Wave  O Out  Vl  Md
KEY SYNC OFF   T'SPOSE C 2      SINE        P Lev  Sn  Sn                          UNDO
VOICE NAME     Wave  01 ♩      Speed  50    6  53   0   0
                               Delay   0    5  57   0   0                          SEND EDITS OFF
PLAY              Note C 3     PMD    29    4  99   2   0
Volume 127     Duration 25     AMD     0    3  53   0   0                          SEND VOICE
OP6 OP5 OP4 OP3 OP2 OP1        PMS     1    2  96   0   0    Number    4
 ON  ON  ON  ON  ON  ON        Sync   ON    1  99   2   0    Feedback  4           END
```



GRAPHIC ENVELOPE EDITOR

ENVELOPE SELECT

6 5 4 3 2 1 P

To use the EDIT voice feature, the voice must be moved into one of the buffers first. The BANK screen is cleared and the EDIT screen appears. All the parameters are clearly visible on the screen and regular voice programmers will quickly be able to familiarize themselves with the arrangement of the principle groupings. There is a GRAPHIC display available for the ENVELOPES and future updates will also include FREQUENCY and KEYBOARD RATE SCALING.

To use the graphic feature one simply grabs the gadget for each level and, holding the mouse button down, move it to the new destination. At the bottom of the screen the LEVEL and RATE values are conveniently displayed. VERY USEFUL.

Real time changes to the parameters are automatically sent and can be checked with the synth or the ST.

The last menu option is RANDOMIZE. I found this feature to be particularly useful. Using combinations of selectable options, including FIXED ROOT, ENVELOPES, FREQUENCY, KEYBOARD SCALING, OUTPUT LEVELS, LFO and a choice of SMALL (10%) or LARGE (30%) Variance, thirty two new voices based on the selected one in the buffer, appear in bank B. Although there are always one or two that sound very similar to each other, depending on the option(s) selected, the voices that have been randomly created are very useable. If, perchance, they do not meet with your approval, simply randomize once more with the same options - or adding/removing others. Or you can replace the voice in the buffer with one selected from one of the new voices in bank B and randomize this!

Apart from the Editing and Randomizing features that have been discussed, the Voice Manager is particularly useful. The ability to load two banks of voices onto the screen and by mouse dragging, selections can be transferred to the other bank - or to the buffer for compiling into a new bank. Each voice slot has been conveniently numbered, to correspond to the DX7 voice panel - slot 1 with switch 1, 2 with 2, and so on. The buffer, of course, is not numbered. The ability to create Performance Banks and to store them easily on disk has made life much more enjoyable!

I have used this program now for over two months and I am more than pleased with it. For me it has been a pleasure to use, although there are a couple of gripes. Perhaps they are due to the language used - Fast Basic from Computer Concepts. These are few in number, but are in important areas.

Firstly, in the voice editing screen, the selecting and changing of data values, either up or down, were slow and sluggish. Using the plus and minus keys were not much better. The procedure for voice naming was also laborious, instead of the little character boxes, a string would have been infinitely preferable, plus all ASCII characters, which in fact, are available. The reason for this is simple - I can get over two hundred banks of sounds on a DSDD disk. This represents over six THOUSAND individual voices - they all need some form of identification and names for some of these wondrous creations quickly run out and some form of classification is needed, RHODES 37 is pretty meaningless! Although the DX7 screen does not display all of these characters - DXpert does, which is important. I use the DX7 screen only as a guide now, DXpert handles all the complex features much more comfortably for me.

Lastly, I would have liked to use some of the GEM attributes that are available - RUBBER BANDING and HIGHLIGHTING when arranging new performance banks. The dragging and precise locating of the relevant voice slot into its destination was very much a hit-and-miss affair and often needed quite a few attempts before transfer was complete.

The manual is clear and very easy to follow. It is housed in a neat, three ring binder - the author also took the trouble to put an On-Screen Help menu.

Incidentally, the voice bank data is compatible with those to be found in the DX folder of PRO 24 from Steinberg. I will periodically be adding disks of banks to the ST Library for use with the DXORGANISER to be found on MDXORG. I have one disk already available, thanks to Laurence - MDXORG VOL1 (1/2 Meg) and by the time the next issue is out, I will have a one meg disk ready (I hope). This will give you over 1700 and 5000 voices respectively.

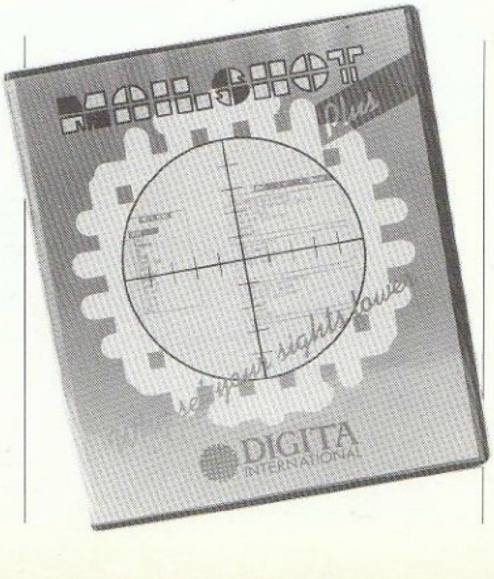
I have no hesitation in recommending this product to all owners of the DX7 series of synths and voice expanders, it represents the most outstanding value for money and is a useful program. I sincerely hope that Laurence will create similar programs for owners of the CASIO, DX9, 2721, FBO 1 and, perhaps, the Roland D-50??

## Mailshot Plus

From Digita International
Price £49.95
Review by Bas Langdor

I think that all of us at some time come into contact with mailing lists, usually at the receiving end. This is often what we refer to as 'junk mail'. However, this is not always so, just think how you got this magazine. For the majority it came through the post, addressed personally to the subscriber. As you can see, a mailing list can be instrumental in bringing the better things in life.

My task today is to tell you about Mailshot Plus, its workings and what uses you could get from it. Firstly, the package contains one program disk, a forty page manual, a quick reference card, some mailing labels and a registration card all encased in a hard plastic case.



Mailshot Plus is totally RAM based which generally means fast sorts and searches but in turn restricts the number of records to be accessed. In a somewhat marked change from the standard 'What you see is what you get' (WYSIWYG) environment, Mailshot Plus is not true GEM based but yet retains the WYSIWYG principle and still uses the mouse to aid movement around the screen.

On booting the program up, the screen is black for mono and dark blue on medium resolution. There is a menu of commands down the left side of the screen and a column of computer mailing labels down the right side of the screen. The actual display is somewhat reminiscent of the early games first available on the ST where you only had a small portion of the screen that actually moved.

The menu has a number of commands: Add, Edit, Remove, Up, Down, Continue, Goto, Layout, Order, Search, Print, File, !O.S. and Quit. Perhaps the easiest way in which to describe all the functions is to go through each menu command. To make this fit in the logical sequence of the menu, I'll follow the same order but may make reference to other commands if deemed necessary. Before we get started, there are some points which are relevant to the operation of the program. The mouse can be used to get around the whole screen and select the menu commands. The cursor keys also allow you to move freely and a few control keys are also operable for such things like inserting and deleting lines or spaces.

So now on to number one - Add. As expected, this lets you add new labels to your list. Type in the information in the same way as you want the label to appear. Press return after each line and twice at the end of each label, or alternately click on 'Add' from the menu. As each label is completed, it scrolls up the screen quite smoothly simulating the movement on a printer tractor feed.

The Edit command lets you alter any label previously stored in the computers memory, again you can move around the label using either the mouse or cursor keys. The remove option can delete any number of labels starting from the default label to any specified number.

The current label is always in the centre section of the label column, consequently, labels can be moved up or downwards. The commands Up and Down move each label individually in the chosen direction. The Continue function scrolls the labels up the screen to the end of the list. To get round the fact that this only works in an upward direction, a Goto feature is included as a separate command. With this, you can go to any label regardless of its direction.

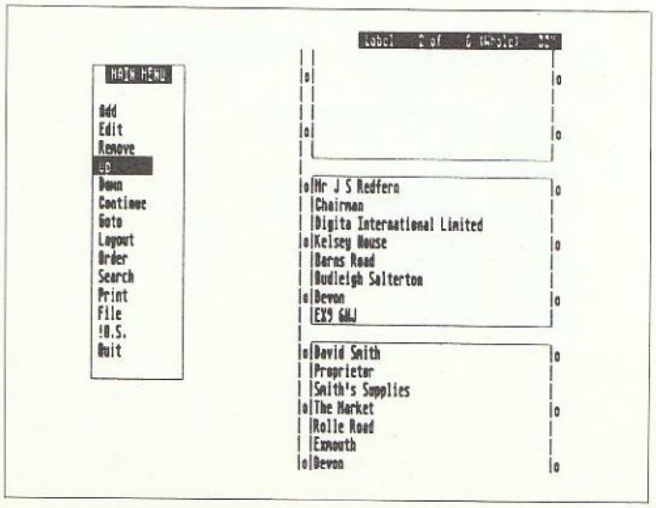
Mailshot Plus doesn't have a design module like many traditional databases, in its place it has a layout function. The default setting is for a 90 x 40 mm label (40 characters x 8 lines). Each label can be up to 48 characters wide and 12 lines deep for data. There are four extra lines for memos. These are not normally printed. Within the layout command, labels can be vertically and horizontally justified to the left/right and top/bottom margins or even centred. Memo lines can be hidden or shown, plus they can also have default values placed in them. In fact the memo lines are quite good, not only do they hold alphanumeric data but also numeric only and date in virtually any format.

When entering data, text can be placed exactly in the way it will appear on an actual label. This is of course a tremendous help when creating labels but can cause problems when it comes to storing the labels in some sort of order. As you input data, it is stored in the position as entered. If the list is going to be any real use, it needs to be able to sort the data over a range of conditions. To keep within the philosophy of WYSIWYG and still have the opportunity to sort by surname or company, etc., then a delightfully handy system of 'Markers' are put into operation.

To see this in perspective, think of creating a label with Title, First Name, and Surname, all on the top line. Normally this would mean that the labels are indexed on Title (Mr/Mrs/Miss/Ms). To get the program sorting effectively, you need to insert a marker before entering the surname. This is easily achieved by pressing the F1 key. From now on, every time you get to entering the surname, just press the F1 key and each surname can be sorted alphabetically. There can be four markers in a label but only one per line. It is quite possible to have the F1 key as the surname marker and the F2 key as a company marker, and so on.

Markers are not printed when the label is produced on the printer, but they can be shown on the screen. It's because of these markers that sorting takes place. Sorting or Order as the menu likes to call it, can be done on any line, Marker or Memo line. Data is sorted alphabetically as a default but if you entered data in a memo either as a numeral or in date format, then providing you sort using the memo lines there's no reason not to display data in numerical or chronological date order.

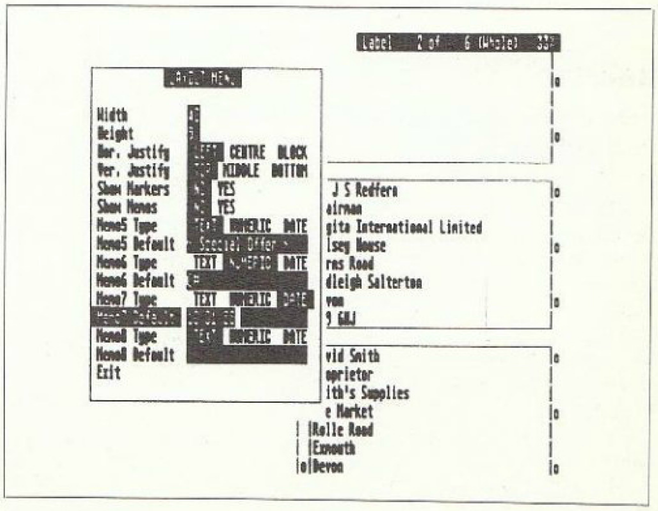
Searching for data is somewhat more comprehensive than the order



Mr J S Redfern
Chairman
Digita International Limited
Kelsey House
Barns Road
Budleigh Salterton
Devon
EX9 6AJ

David Smith
Proprietor
Smith's Supplies
The Market
Rolle Road
Exmouth
Devon

Entry Screen



J S Redfern
airman
gita International Limited
lsey House
rns Road
dleigh Salterton
von
9 6AJ

vid Smith
oprietor
ith's Supplies
e Market
Rolle Road
Exmouth
Devon

Layout

option. This section has three parts, titled Location, Condition and Requirement. The easy way in which to search for data is to type in the location command 'Any', the condition (operator) ' = ' and the 'requirement', i.e. whatever you are looking for. Again to put this into perspective, think of searching your file for all the labels with London. By typing in the command 'Any = London', the computer will search all the lines for the word London. This will also pick up words like Londonderry. To restrict this from happening all you need to do is ensure that there is a space either side of London. Faster searches can be achieved by indicating on which line to search. In real practice, you're only talking about a few seconds difference. To search 1500 records took 9 seconds on any line and only 5 seconds on a specific line.

To make better use of the search facility, apart from the standard operators of Equal to, Not Equal to, Greater Than, Less Than, etc. the conditional operators of AND/OR/NOT have been included. This means that you can link several search criteria together.

Now whenever you extract data, the result is either sent to the screen or alternately to a subset. This subset can be 'printed' either to the printer, or to disk in ascii format. The subset can also be saved to disk which can be used as a separate file. Data can also be edited and removed from the subset and this will not affect the whole file. If you decide to create another subset from the original subset, then the most recent one will replace the existing one in memory. Finally in the search command there is an option to detect duplicate labels.

Printing can be done in three different ways, Print Labels, Print Summary and Print to Disk in ascii format. Mailshot Plus is configured to a standard DOS text printer which is compatible with the majority of printers around. There is however a printer initialisation command available for changing the text into any style such as condensed or emphasized for example. Labels can be up to 9 across and there is a left margin command for the first label only. To use more than one label across means some nifty setting up in the layout option. Mailshot Plus can print up to 999 copies of a single label.

To get a print run, you will need to search for your selection or work from a subset otherwise the whole file will be printed. User defined messages can be printed out on the bottom line of every label. This can be used for standard items such as 'Special Offer' or 'Important News'. Whatever your message you have a maximum of 48 characters only. To selectively define a group of labels for a special message, you need to place a special string code on a line for every label that requires the message. For instance, you could tag certain labels with the code and enter the special message in the string option from the print menu.

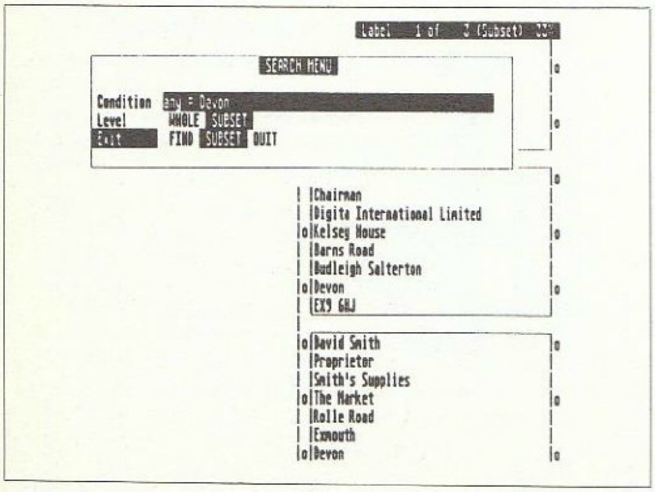
Sequential numbers can also be placed on the labels. These are user defined and can start or end at any position. Printing labels to disk is easy enough, just type in a file name and extension in the print menu command line and click on the file command.

Sending information to labels is one thing but for many users knowing what's been printed out is all important. There is a way to do this, the Summary command. Once again, select this from the print menu, choose your line numbers, markers, memo lines and click on the summary command. Unfortunately you cannot use the number or string feature in the summary output but it is possible to increment the summary entries from number 1.

Finally, a look at the loading and saving of records. Files can be loaded and saved in the normal label format. This also stores the markers and memo lines. When loading and saving ascii files, a line separator and a record separator need to be inserted. Mailshot Plus already uses control j as the default line separator and control j, control j for the record separator. For records that are too long for the label format, you can either truncate or split the lines.

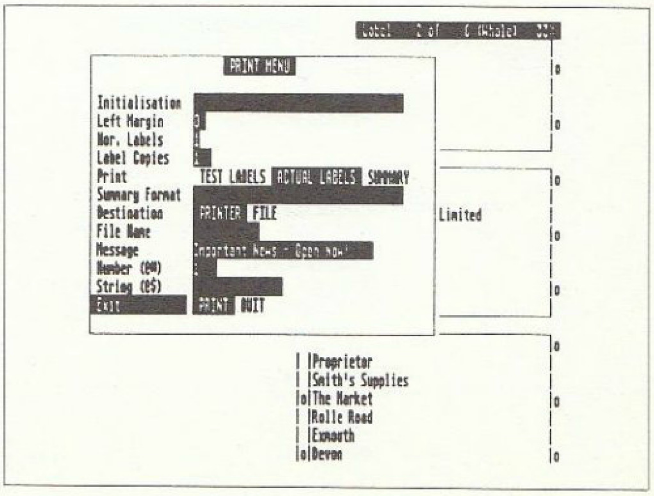
Truncate, as the name implies will simply cut off the characters that overhang the prescribed line length. Split on the otherhand is an intelligent system and will find a suitable break in the line and place the remaining text on the next line. Using the ascii load and save feature has one drawback, markers and memo lines cannot be saved with the label information.

Label files can be merged together to create larger ones, in reverse, large files can be made smaller by creating subsets. There is a small menu command called !O.S., this is a sort of MSDos shell. At the moment it's only function is the DIR (see directory) command. The final command is Quit, this takes you back to the desktop after a confirmation sequence.



```
Label   1 of   3 (Subset)

SEARCH MENU
Condition  Any = Devon
Level      WHOLE SUBSET
Exit       FIND SUBSET QUIT

|Chairman
|Digita International Limited
|Kelsey House
|Barns Road
|Budleigh Salterton
|Devon
|EX9 6HJ

|David Smith
|Proprietor
|Smith's Supplies
|The Market
|Rolle Road
|Exmouth
|Devon
```

Search



```
Label   2 of   6 (Whole)

PRINT MENU
Initialisation
Left Margin     0
Hor. Labels     1
Label Copies    1
Print           TEST LABELS ACTUAL LABELS SUMMARY
Summary Format
Destination     PRINTER FILE
File Name
Message         Important News - Open now!
Number (@#)     1
String (@$)
Exit            PRINT QUIT

Limited

|Proprietor
|Smith's Supplies
|The Market
|Rolle Road
|Exmouth
|Devon
```

Print

Right, we're at the end of this review and what's the conclusion. Mailshot Plus works well as a stand alone mailing program. Even though the publishers state it can be used for a multitude of other projects because of the limitations of one file, one subset and average reporting facilities, it's not really up to any more than the original intention.

Saving and loading files take an eternity, nearly three minutes to load in 1500 records, the saving took even longer. The search procedure, whilst fairly impressive, is restricted because of the one subset rule and the fact that there is no way that any search criteria can be stored on disk. It would have been handy to have a logical IF/THEN procedure and the inclusion of a Global Update would be useful. The reporting procedure could do with a little more work doing to it, such as headers, titles, page numbers and perhaps a simple pre-processor for letters could make this package very attractive.

On the positive side, Mailshot Plus is easy to use and understand, the WYSIWYG principle is reassuring. The speed of searching and sorting is impressive. I liked the system of markers and the goto function. The manual is friendly and very helpful not just in the program operations but also about mailing lists and their functions. There's 60 days free written and telephone support, plus the option to join the maintenance support which provides free program updates for £25 per year.

My final conclusion is that for any business or club secretary, Mailshot Plus could be a good buy. The program retails at £49.95 and if it were reduced in price or the above amendments put into operation it may be well worth looking at for the enthusiast.

## Chessbase

**Author: Matthias Wullenweber.**
**Distribution: Saitek Industries Ltd., 4 Bridge Studios, 318 - 326 Wandsworth Bridge Road, London SW6 2TZ.**
**Price: Numerous options outlined in article.**
**Reviewer: Mike Stringer.**
**ST 1/2 Meg (limited), 1 Meg (desirable) MONO ONLY.**

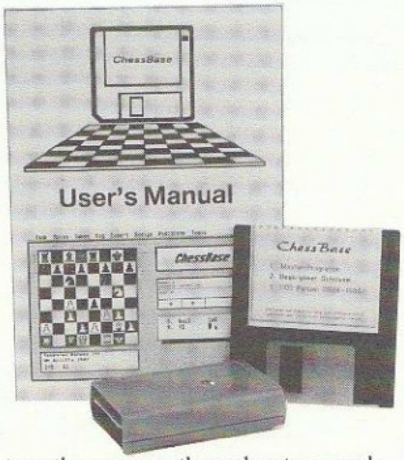
CHESSBASE is not a game - but you can play chess games with it, this is a chess database. The basic system comes with an illustrated manual, a ROM cartridge and the program disk containing the main program, over 1000 chess games and a KEY with over 150 popular openings.

A database for chess I hear you ask? Of what possible use is a chess database?

In many respects chess can be likened to golf or snooker. To play these games well, you must practice and when the practice is over you practice some more. The greats in these events, such as Nick Faldo and Steve Davis, practice for hours before a match, play the match and immediately return to the practice table.

Chess also requires practice. The Grand Masters will study and analyse hundreds of games of their opponents, searching for information. This information relates to their strengths and weaknesses in various circumstances.

In similar circumstances, the Generals at the battle field will collect information from spies and observers on his enemies battle strength - numbers and position, etc. Make no mistake, chess is a battle, the pieces are equally balanced and the board is the field of battle. If you can gain an insight into your opponent's strength or some other advantage, such as a certain sequence of moves at a given point in a game, you will stand a much better chance of winning.



CHESSBASE will give access to such information, very rapidly, from games that have been painstakingly transferred from sources such as books, magazines, clubs and archive material. It is possible to store over 6000 games on a standard diskette and it is anticipated that a CD ROM could store every known game! This is food for thought, after seeing the new CD at the recent Atari Autumn Show!

For a very modest cost, CHESSBASE are providing a most impressive back-up facility for registered owners - including a bi-monthly magazine and numerous disks including one that has all the World Championship Games from 1886 to 1986!

The Soviets are so impressed with the product, that many little tovarichs are busy transcribing over 300,000 games in their library over to CHESSBASE. Whether these files will be made available to us, remains to be seen.

On booting up the disk, after inserting the cartridge into its port, before switching on, a RAM disk is created automatically (Drive F). For my own purposes, I also created a drive icon for my second drive using the normal method. There is only one disk icon, plus the RAM icon, shown on the desk top.

Each CHESSBASE database file is composed of four related files, these are quickly copied into the RAM disk. Once done, the main program can then be booted.

CHESSBASE is GEM based, the main menu options are to be seen at the top. Because many readers will not be familiar with the in-depth structure of chess protocol, the contents of some of these drop-downs will only be briefly covered. Those of you who are familiar, will recognize the power that is available!

Newcomers need not be put off because the CHESSBASE manual does give an extremely good description of all the features and a very detailed coverage of Opening Classifications and Sub-classifications.

The headings are: MOVES, GAMES, KEY, EXPORT, ASSIGN, POSITIONS and TOOLS. Most features can be activated in three ways: MOUSE, KEYBOARD and MENU. For brevity, I will only give the menu commands.

### MOVES

Take Back: Once a game has been loaded and you are some way into it, you will often need to backup a move, this option performs that task.

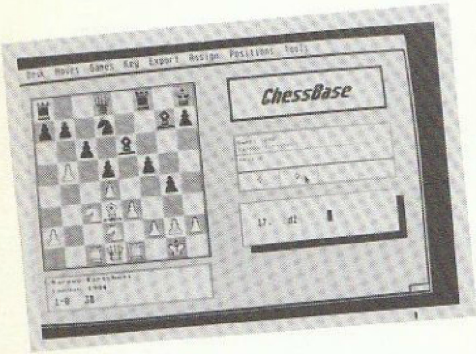
One Move: Advances one move.

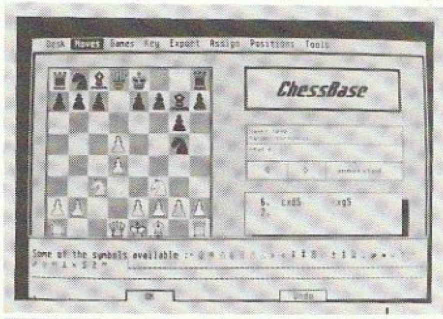
Move Number: Similar to a Go-To-Line Number feature in a Word Processor.

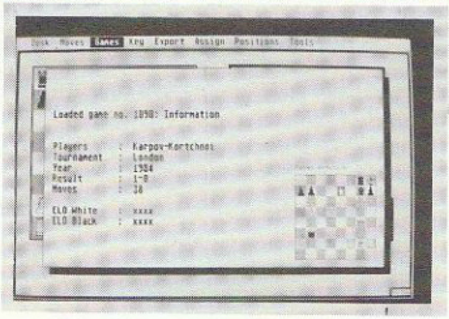
Flip Board: Rotates the board through 180 degrees.

Leonardo: SAITEK market a very nice chess board that can be plugged into the ST serial port. This option allows the necessary communication paths to be set up.

ase
l It lea
nce it h
ve f

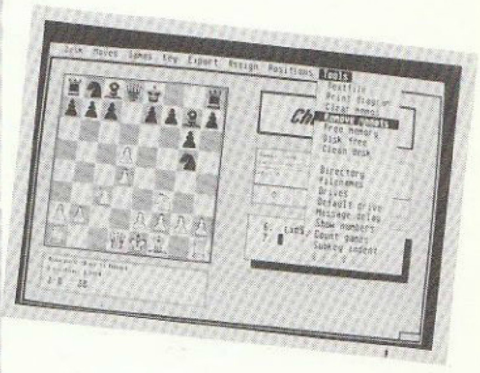



Desk Moves Games Key Export Assign Positions Tools
ChessBase
Some of the symbols available



Loaded game no. 1898: Information

```
Players     : Karpov-Kortchnoi
Tournament  : London
Year        : 1984
Result      : 1-0
Moves       : 38

ELO White   : xxxx
ELO Black   : xxxx
```



Annotation: A feature to allow description of every move. There is sufficient space to enter 255 characters and, by pressing the CAPSLOCK key, a redefined character set is activated. Certain letters now become INFORMATOR and NEW IN CHESS symbols. This feature allows quite detailed notes to be taken with the minimum amount of valuable RAM usage.

Evaluation: This allows brief comments to be shown in the window that contains the algebraic moves.

!? etc.: The above relates to positions, this feature relates to moves. '!' traditionally means a good move, '?' a doubtful one.

## GAMES

List and Select: This is a very powerful search feature. Upon selection, a box appears which allows you to search by PLAYER, YEAR and KEY. For example you can search the database for '-Nunn', '1985', 'Grunfeld'. Clicking OK, the database will now search for all games where Dr John Nunn, playing BLACK in a game with a Grunfeld opening! You could even sub-divide the option by specifying a given opponent!

By clicking only on the OK button, all the games in the database will be listed. Hitting any key pauses the listing and any game can be loaded simply by highlighting it and double clicking.

Games that have been retrieved by the search commands can be posted to a special file for further processing by selecting EXPORT. A TEXTFILE of the game number, contestants, year, event, moves number, is created in a special file which can be edited by a word processor or printed from the desktop.

Save: If you have entered a new game, or have modified an existing game, this option allows it to be added to the database.

Load: Allows you to load a game from its record number.

Buffer: Stores the current game into a special buffer for immediate retrieval. Useful when comparing games.

Get Buffer: Retrieves Buffered game.

New Game: Initializes the board.

Edit Game: Game data made available.

Delete: Deletes a game from the database. Used with caution.

Delete Many: As above, but more so!

## KEY

I mentioned earlier that some of the menu options involved a comprehensive knowledge of chess opening classifications, the KEY option uses this knowledge extensively.

Main Level: Selecting an opening, a list of the sub-levels will also appear. Clicking on one of these, you may find further levels nested.

More Levels: This option only displays the sub-levels.

Contents: Allows you to load games from the KEY number.

Overview: Allows you to select the depth of the level of a KEY.

Position: Displays positions linked to the KEY.

New KEY: Allows you to create a new KEY.

New Sub-KEY: Allows you to create a new sub-KEY.

Edit KEY: Allows you to re-name a KEY.

Move sub-KEY: This is used for expanding and improving an opening structure.

KEYFILE II: Initiates a second KEY.

## EXPORT

This menu allows games from other databases to be incorporated into the current, or new one. Most of the following options are inactive until the program has been informed of the name and location of the file that is to be worked on.

Start Export: Allows you to give the necessary information.

List Export: Allows you to select games from the destination file.

Export One: This adds the exported game to the main file.

Export Many: Adds many exported games to the database based on the record number.

KEY contents: This allows you to pre-sort the file and export games based on a KEY or sub-KEYs.

Erase Export: Deletes the export file.

Import: This allows you to import games without classifying them.

Next Game: Clicking on this option automatically loads the next game from the export file. VERY USEFUL.

## ASSIGN

This menu controls all the manual and automatic commands for the classification of games.

Identify: A most useful feature which describes the opening, including any sub-levels and creates a list of games from the database similar to it.

Import and Assign: This feature automatically classifies export files according to the openings KEY.

Re-Assign All: An awesome command that deletes the current opening KEY and re-classifies the whole database on a new KEY.

Assign by Hand: Allows you to assign a game manually.

Remove by Hand: The opposite of the above.

## POSITIONS

These commands assist CHESSBASE to maintain the library.

Search: This feature searches the .POS file for the current board state for a match.

List All: Lists the .POS file graphically and the KEY with which it is associated.

New Position: This feature adds a new position into the .POS file.

KEY Link: Allows you to modify a link to a KEY.

## TOOLS

Textfile: One of the remarkable features of CHESSBASE is the compression of data. A single move has been reduced to 1 byte! In order to give a text listing of a game(s), this data needs to be processed.

Print Diagram: This produces a graphics dump with an Epson printer.

Clear Memos: Self explanatory.

Remove Annots: Removes all annotations and explanations from the current game.

Free Memory: RAM check.

Disk Free: Checks disk for available space.

Clean Desk: A useful feature and it is the first program that I have seen it in. This command clears any GEM 'cock-up' (a computer technical term), caused by screen re-write bugs.

File Names: Self explanatory.

Drives: Allows you the option of changing drives.

Default Drive: Allows you to define the default drive for porting files such as textfiles.

Message Delay: All warning and prompt flags are on the screen for a set period of time. This option allows you to re-set this value.

Show Numbers: This feature will display the record numbers of KEYS when printing to the screen.

Count Games: This is a toggle switch for CHESSBASE to count the number of entries in any given KEY or sub-KEY. This will slow the game down - hence the option to switch it off.

Sub-KEY Indent: Allows you to arrange the sub-KEY printing to your taste.

Define Piece Names: Allows you to change the chess symbols.

This completes the brief tour of the menu and their commands. Although I have tried to keep such explanations/descriptions brief, you can appreciate I am sure, the amount of care and thought that the author has put into the program. If you consider the complexity of the subject, these options make it VERY USER FRIENDLY. Do not forget that most of these commands can also be accessed through mouse commands and also the keyboard.

## GENERAL IMPRESSIONS

First of all, let us examine the manual. In many products this part of the package is very disappointing. Many companies would rather have fancy little boxes/folders complete with 'artist impression' artwork to tart up the product. Not so with the CHESSBASE manual. It is superb. Nothing has been left to chance. Considering the product is German, the translator must be given a pat on the back. The English is first class. There were no spelling mistakes or grammatical errors that I could find.

There were two nice touches. First, there are little stars (asterisks). These little warnings inform you, if inexperienced, that possible dangers are lurking within that command. The greater the number of stars, the greater is the damage that can occur IF CARE IS NOT OBSERVED.

Secondly, you frequently come across little boxes. These contain resumes, advice, hints and wrinkles. They are definitely well worth reading.

Although it is only 71 pages long, it is crammed with information. The illustrations are mostly screen shots or printer dumps.

Of the program itself, I have already mentioned the user-friendliness. The graphics are superb. Very clear and crisp, thanks to the choice of high resolution.

The speed of searching and screen displays are breathtakingly fast. To play a move I prefer to use the mouse and this is pointed to the right arrow box. The action is so fast that, at first, I had to install the CONTROL desk accessory to slow the mouse down! I found that I could not gauge the click duration to persuade the program to move on just one move! And this is after three years experience with the ST!

The moves of an average game (say 40 moves) are paged through in about 3 seconds if the mouse button is held down. Apparently, Gary Kasparov plays all new games AT THIS SPEED to appraise it!! Searches are also very fast, depending on the filtering options chosen. The current search record number is a blur in the top left hand corner.

When a game is selected for display the program momentarily displays a very useful box. This contains the essential features of the record: name of contestants, date, tournament, number of moves, the opening KEY and a miniature chess board of the final position. Clicking on OK clears the box and the main board is set up.

Each CHESSBASE is configured individually to the purchaser. I believe that all major languages are catered for - including Chinese and Japanese!

The acclaim that the product has received from the Professional players is unanimous. The CHESSBASE Magazine is edited by one of our Grand Masters Dr John Nunn.

To say that I am impressed with this product is an understatement! Did I find it of any use? Considering the short time that I have been using the program, I feel that the strength of my own game potential, both in offensive and defensive play has increased by about 25 percent.

I have managed to persuade our Editor that chess is a very popular pastime with many computer owners and that we will have a regular CHESSBASE column in MONITOR. I know of no other product that has sold so many STs mainly for this purpose. MIDI applications were clearly in the lead, but they have already been left behind! I hope, in due course, to include disks of databases in the ST Library!

Now to the price options that are available. To start off there is the STARTER PACKAGE.

This particular package has been targeted to chess players who are uncertain if they need such a powerful program. The package comes complete with 250 games and is capable of playing ALL CHESSBASE GAMES and costs a modest £24.95! Most of the reference system is missing, but it does include the high resolution graphics, input, storage, replay and Leonardo.

CHESSBASE with the ROM cartridge, over 1000 games, 150 KEY openings and manual costs only £129.95.

The third option available is the 'professional package'. This includes, in addition to the main package, a 6 issue subscription to the CHESSBASE Magazine, the INFORMATOR KEY with 1000 positions, the NEW IN CHESS KEY with 1500 positions, NEW IN CHESS Volume 5 with over 1500 games, the World Championship Games 1886 - 1986, the German Chess League 1986/1987 containing 1000 games.

The CHESSBASE Magazine is

disk-based and contains about 1000 current games, news, tools, updates, new functions and items of general interest. There is a special price of only £199 for this pack. A saving of £40 over normal prices. That may seem expensive, but a great deal of blood, sweat and tears has gone into these products! Actually it is £150 cheaper than my set of golf clubs! I still have my green fees to pay, plus the balls at £1.60 a time!

CHESSBASE is a product that will give years of pleasure - one would not get tired of it and, at the same time, increase your potential in this fascinating game.

The World Champion, Gary Kasparov was the first person to have a copy and he has put his endorsement on the product by stating at a press conference: 'This is the most important development in Chess Study since the invention of printing'. Who am I to argue?

I have no hesitation in declaring CHESSBASE THE PRODUCT OF THE YEAR FOR 1987.

## Lattice C V3.04

From Metacomco
Price £99.95
Reviewed by Keith Mayhew

Lattice C has been reviewed in these pages before, in a comparison with Megamax C. This review covers the new version 3.04 of Lattice, which is available at the same price as the older version, or as an upgrade to existing owners for £34.50.

The main criticisms before were that although the compiler was excellent, the package lacked a resource editor and the manual could easily fall apart!

It is very encouraging to see that Metacomco are not only offering a new version of the Lattice compiler but have upgraded the rest of the package in an attempt to fight off the competition.

The major changes include a new manual (in excess of 600 pages and does not fall apart any more!), a new set of very extensive library files, faster floating point, more code optimisations, an upgraded editor, the Make utility, the K-Resource editor and a symbolic debugger.

The really surprising part is that the cost of the product has not changed, despite the inclusion of two other packages: K-Resource and Make. Both have been reviewed in Monitor before. K-Resource is probably the best resource editor available and Make is a very useful utility for aiding the development of large programs. For these reasons alone the new Lattice package is much better value than other comparable compilers available. Any existing owner who is wondering if it is worth upgrading should realise that the upgrade cost is less than the price of K-Resource alone - so the answer is definitely YES!

The manual is well presented with an adequate description of each of the components of the package including a quick introductory tutorial on how to compile and link a program under Lattice. My complaint here is: a step-by-step guide has been given to compiling a sample program, but it does not work! The reason turned out to be that the files are grouped in folders on the disks and the header files are in one folder when they need to be in the same folder as the compiler! Not exactly a major problem but how did it happen?

The majority of the manual (over 400 pages) is devoted to detailed descriptions of a truly vast collection of routines from different environments: UNIX, XENIX, LATTICE, ATARI and ANSII. In fact the libraries must be the most comprehensive supplied with any C compiler. Routines are included for string searching with pattern matching, set inclusion for characters, token parsing, environment access and many other diverse and useful functions rarely found elsewhere. A very annoying feature of the manual is that it is not always possible to find a specific function without searching through a LOT of pages. The reason for this is that some functions are grouped with others under their title because they are similar in operation - the index is not complete enough to solve this random access problem!

For an unknown reason, Metacomco advertise that GEMDOS, BIOS and XBIOS libraries have documentation: I have yet to find any of the functions in the manual although the actual system calls are documented and the necessary include file for their use is supplied. Furthermore, direct access to the Line-A graphics routines is provided from C which is a very useful facility.

The compiler has been further improved to remove any known bugs and produces slightly faster code than before. The main change is the inclusion of the ANSII proposals for a standard C. This provides a type 'void' for functions which do not return a value and 'enum' for enumerations as in Pascal/Modula II. The other great advantage is the ability to type check arguments to functions. If the ANSII features disagree with your existing source then a flag is provided which will compile to the K & R definition.

The editor, although repackaged as a GEM application, is still the same editor as has always been supplied. Whilst it is easier to use than the old version it is not an excellent GEM editor. Its main problems are that it is slow at scrolling, it does not allow text to be directly moved between windows (a file has to be used) and some of its operations are extremely slow.

One of the best, and completely new, parts of this package is the inclusion of an excellent debugger called debug plus. It has been designed so that it is particularly easy to use to study C programs by allowing the use of symbols and stack traceback facilities. It allows you to find the first symbol reference before a certain location so as to determine which routine you are inside. Its macro facility allows any command to consist of any number of other existing commands or macros, thus enabling the user to build a set of custom and fairly powerful commands. The manual provides a good example of 'hunting the bug' in its description of debug plus, which gives a taste of its usefulness. To help provide a good development environment, debug plus can also be used for linking C object files in memory. Thus the link is not only significantly faster but the debugger is still at hand for testing. GST's linker is still supplied so that a stand-alone program file can be produced.

For the price/performance of this package it is certainly going to be hard to beat. It has many excellent features but lets itself down with some fairly minor flaws. Megamax C has only one real advantage over Lattice though and that is its one-pass compiler. The price advantage of Lattice together with K-Resource should make it a very attractive proposition to programmers, whether professional or just starting.

# MAIL ORDER ATARI ST SOFTWARE

| Title | RRP | Our Price | Title | RRP | Our Price |
|---|---|---|---|---|---|
| Addictaball | £14.99 | £12.75 | Airball | £24.95 | £21.25 |
| A/ball Constuct. Set | £14.95 | £12.95 | Altair | £19.95 | £16.95 |
| Alternate Reality | £24.95 | £21.25 | Arkanoid | £14.95 | £12.75 |
| Arena | £24.95 | £21.25 | Autoduel | £24.95 | £21.25 |
| Amazon | £19.95 | £16.95 | A Mind F'er Voyaging | £34.95 | £29.75 |
| Backlash | £19.95 | £16.95 | Barbarian | £24.95 | £21.25 |
| Battlezone | £29.95 | £25.50 | Barbarian, Ult. Warrior | £14.99 | £12.75 |
| Boulderdash Cons. Set | £24.95 | £21.25 | Brataccas | £24.95 | £21.25 |
| Balance of Power | £29.95 | £25.50 | Bridge Player 2000 | £19.95 | £16.95 |
| Brian Clough's F/ball | £24.95 | £21.25 | Borrowed Time | £24.95 | £21.25 |
| Bally Hoo | £29.95 | £25.50 | Breakers | £44.95 | £37.95 |
| Brimstone | £44.95 | £37.95 | Bureaucracy | £34.95 | £29.75 |
| Crafton & Xunk | £19.95 | £16.95 | Crystal Castles | £19.99 | £16.95 |
| Cards | £19.95 | £16.95 | Check Mate | £14.95 | £12.75 |
| Chess | £24.95 | £21.25 | Colonial Conquest | £29.95 | £25.50 |
| Crimson Crown | £19.95 | £16.95 | Cutthroats | £29.95 | £25.50 |
| Champ. Basketball | £24.95 | £21.25 | Champ. Football | £24.95 | £21.25 |
| Champ. Wrestling | £24.95 | £21.25 | Death Strike | £14.95 | £12.75 |
| Deep Space | £24.95 | £21.25 | Delta Patrol | £24.95 | £21.25 |
| Diablo | £29.95 | £25.50 | Deadline | £29.95 | £25.50 |
| Eagles Nest | £19.95 | £16.95 | Extensor | £19.95 | £16.95 |
| Extravaganza | £12.94 | £10.95 | Electronic Pool | £19.95 | £16.95 |
| Eden Blues | £24.95 | £21.25 | Enchanter | £29.95 | £25.50 |
| Essex | £44.95 | £37.95 | Fire Blaster/War Zone | £14.95 | £12.75 |
| Flip Side | £19.95 | £16.95 | F-15 Strike Eagle | £24.95 | £21.25 |
| Flight Simulator II | £49.95 | £42.50 | Fahrenheit 451 | £19.95 | £16.95 |
| Forbidden Quest | £39.95 | £33.95 | Gauntlet | £24.95 | £21.25 |
| Goldrunner | £24.95 | £21.25 | Gato | £29.95 | £25.50 |
| Great Battles | £34.95 | £29.50 | Golden Path | £19.95 | £16.95 |
| Gateway | £49.95 | £42.50 | Guild of Thieves | £24.95 | £21.25 |
| Hades Nebula | £19.95 | £16.95 | Hex | £39.95 | £33.95 |
| Harrier Strike Mission | £49.95 | £42.50 | Hardball | £24.95 | £21.25 |
| Hacker | £24.95 | £21.25 | Hacker II | £24.95 | £21.25 |
| Hitch Hiker Guide | £29.95 | £25.50 | Hollywood Hijinx | £29.95 | £25.50 |
| Ind. Jones/Temp.Doom | £19.95 | £16.95 | InternationalKarate | £19.95 | £16.95 |
| Infidel | £29.95 | £25.50 | Joust | £29.95 | £25.50 |
| Jupiter Probe | £14.95 | £12.75 | Jewels of Darkness | £19.95 | £16.95 |
| Karate Kid II | £24.95 | £21.25 | Karate Master | £12.95 | £10.95 |
| Kings Quest II | £24.95 | £21.25 | Knight Orc | £19.95 | £16.95 |
| Liberator | £12.95 | £10.95 | Lands of Havoc | £19.95 | £16.95 |
| Leaderboard | £24.95 | £21.25 | Leadbd. Tourn. Disk 1 | £ 9.95 | £ 8.95 |
| Lurking Horror | £29.95 | £25.50 | Macadam Bumper | £24.95 | £21.25 |
| Major Motion | £19.95 | £16.95 | Mercenary Compendium | £24.95 | £21.25 |
| Metro Cross | £24.95 | £21.25 | Metropolis | £12.95 | £10.95 |
| MGT | £24.95 | £21.25 | Mission House | £19.95 | £16.95 |
| Monkey Business | £24.95 | £21.25 | Mouse Trap | £14.95 | £12.75 |
| Mudpies | £19.95 | £16.95 | Mindshadow | £24.95 | £21.25 |
| Mortville Manor | £24.95 | £21.25 | Mindwheel | £44.95 | £37.95 |
| Moonmist | £29.95 | £25.50 | Mean 18 | £29.95 | £25.50 |
| Mean 18 Famous Courses | £19.95 | £16.95 | Ninja Mission | £ 9.95 | £ 8.95 |
| Nine Princes in Amber | £19.95 | £16.95 | Nord & Bert | £29.95 | £25.50 |
| Outcast | £ 9.95 | £ 8.95 | Oo-Topos | £19.95 | £16.95 |
| Plutos | £14.95 | £12.75 | Prohibition | £19.95 | £16.95 |
| Pinball Factory | £24.95 | £21.25 | Passengers on the Wind | £24.95 | £21.25 |
| Phantasie | £24.95 | £21.25 | Phantasie II | £24.95 | £21.25 |
| Phantasie III | £24.95 | £21.25 | Pirates of Barbary C | £12.95 | £10.95 |
| Pawn (The) | £24.95 | £21.25 | Perry Mason | £19.95 | £16.95 |
| Pludered Hearts | £29.95 | £25.50 | Planetfall | £29.95 | £25.50 |
| Red Alert | £19.95 | £16.95 | Roadwar 2000 | £24.95 | £21.25 |
| Rogue | £24.95 | £21.25 | Screaming Wings | £14.95 | £12.75 |
| Slap Fight | £19.95 | £16.95 | Space Pilot | £19.95 | £16.95 |
| Spy vs Spy | £24.95 | £21.25 | Star Raiders | £19.99 | £16.95 |
| Starglider | £24.95 | £21.25 | ST Protect./Space Station | £24.95 | £21.25 |
| ST Wars | £39.95 | £33.95 | Super Cycle | £24.95 | £21.25 |
| Shanghai | £24.95 | £21.25 | Scenery Disk 7 (SIM) | £24.95 | £21.25 |
| Scenery Disk 11 (SIM) | £24.95 | £21.25 | Shuttle II | £24.95 | £21.25 |
| Silent Service | £24.95 | £21.25 | Strike Force Harrier | £24.95 | £21.25 |
| Super Huey | £19.95 | £16.95 | ST Karate | £19.95 | £16.95 |
| ST Pool/Shuffleboard | £19.95 | £16.95 | Super Tennis | £24.95 | £21.25 |
| SDI | £29.95 | £25.50 | Sentinel | £19.95 | £16.95 |
| Sidewalk | £19.95 | £16.95 | Space Quest | £24.95 | £21.25 |
| Star Fleet 1 | £55.00 | £46.75 | Sundog - Frozen Legacy | £29.95 | £25.50 |
| Sword of Kadesh | £19.95 | £16.95 | Silicon Dreams | £19.95 | £16.95 |
| Spiderman | £19.95 | £16.95 | Seastalker | £29.95 | £25.50 |
| Sorceror | £29.95 | £25.50 | Spellbreaker | £29.95 | £25.50 |
| Starcross | £29.95 | £25.50 | Stationfall | £29.95 | £25.50 |
| Suspect | £29.95 | £25.50 | Suspended | £29.95 | £25.50 |
| Terrestial Encounter | £14.95 | £12.75 | Terrorpods | £24.95 | £21.25 |
| Time Bandit | £29.95 | £25.50 | Tonic Tile | £19.95 | £16.95 |
| TNT | £19.95 | £16.95 | Tracker | £24.95 | £21.25 |
| Trail Blazer | £24.95 | £21.25 | Turbo GT | £15.95 | £13.50 |
| Typhoon | £19.95 | £16.95 | Techmate Chess | £19.95 | £16.95 |
| Tee Up | £14.95 | £12.50 | Thai Boxing | £14.95 | £12.50 |
| Tai-Pan | £19.95 | £16.95 | Temple of Apshai Trilogy | £24.95 | £21.25 |
| Tass Times in Tone Town | £24.95 | £21.25 | Transylvania | £19.95 | £16.95 |
| Treasue Island | £39.95 | £33.95 | Trinity | £34.95 | £29.75 |
| Ultima 2 | £24.95 | £21.25 | Ultima 3 | £24.95 | £21.25 |
| Universe II | £69.95 | £59.95 | Vegas Gambler | £24.95 | £21.25 |
| Wanderer 3D | £24.95 | £21.25 | Word for Word | £39.95 | £33.95 |
| Winter Games | £24.95 | £21.25 | World Games | £24.95 | £21.25 |
| Wishbringer | £29.95 | £25.50 | Witness | £29.95 | £25.50 |
| Xevious | £24.95 | £21.25 | Zkul/West | £24.95 | £21.25 |
| Zork I | £29.95 | £25.50 | Zork II | £29.95 | £25.50 |
| Zork III | £29.95 | £25.50 | Q Ball | £19.95 | £16.95 |

Educational & Childrens ST Software:

| Title | RRP | Our Price | Title | RRP | Our Price |
|---|---|---|---|---|---|
| Decimal Dungeon | £39.95 | £33.95 | Donald Duck Playground | £24.95 | £21.25 |
| Election Program (The) | £29.95 | £25.50 | First Shapes | £49.95 | £42.50 |
| Fraction Action | £39.95 | £33.95 | Intro to ST Logo | £19.95 | £16.95 |
| Kid Talk | £49.95 | £42.50 | Little Computer People | £34.95 | £28.95 |
| Math Talk | £49.95 | £42.50 | New Tech Col Book | £19.95 | £16.95 |
| Perfect Match | £ 9.95 | £ 8.95 | Read & Rhyme | £39.95 | £33.95 |
| Speller Bee | £49.95 | £42.50 | Trivia Challenge | £19.95 | £16.95 |
| Typing Tutor | £34.95 | £28.95 | Winnie the Pooh | £24.95 | £21.25 |

Full range of Accounts, Art, Music, Desktop Publishing, Spreadsheets, Databases, Utilities available at discount prices. Phone 0432 50836 with your requirements and we will be glad to quote you a price!
All prices are fully inclusive. P & P free!

| Title | RRP | Our Price | Title | RRP | Our Price |
|---|---|---|---|---|---|
| Fast ASM | £19.55 | £16.50 | Fast ST Basic (ROM) | £89.90 | £76.50 |
| Fast ST Basic (Disk) | £44.85 | £37.95 | MCC Macro Assembler | £49.95 | £42.45 |
| VIP GEM | £228.85 | £194.50 | Typesetter Elite | £34.95 | £29.50 |
| Back Pack | £49.00 | £41.50 | Back Up | £24.95 | £21.25 |
| Disk Doctor | £29.95 | £25.45 | Graphic Toolbox | £69.95 | £59.45 |
| Label Master | £39.95 | £33.95 | PC Ditto | £49.95 | £42.45 |
| Utilities | £39.95 | £33.95 | Micro Time Clock Card | £34.95 | £29.75 |

Many more software titles avaiable at discount prices plus Atari and Third Party hardware. Phone 0432 50836 and we will be glad to quote you a discount price. Remember, all prices are fully inclusivo. Postage and packing is FREE!

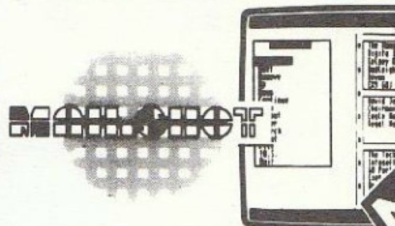
Please make cheques or postal orders payable to: **J. R. DAVIS (COMPUTER SUPPLIES) 10 INGRAM AVENUE, HOLMER, HEREFORD HR4 9RD. Tel: (0432) 50836**

Atari 8 Bit users! We offer comprehensive discounts on XL/XE software - Please phone.

We supply only brand new & original products. All items sold subject to availability.

# ATARI® ST RANGE

1



NEW

## MAILSHOT ... £24.95
## a revolutionary product

A powerful menu driven mailing list program which uses a unique system for on-screen scrolling of labels. This WYSIWYG (What You See Is What You Get) system means that any label format you define on the screen will be identical when printed.

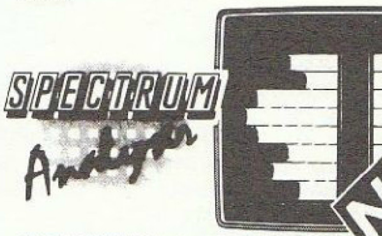
As well as powerful sorting and searching (search for anything, anywhere!), **Special Routines** include: detection of duplicate labels, surname sorting and many, many more.

**FLEXIBILITY** - you may also use MAILSHOT for many other routine day to day labelling and database applications.

## MAILSHOT PLUS £49.95

This advanced version of Mailshot is simply the MOST powerful and flexible mailing program available. Ask for our brochure for full details.

2



NEW

## SPECTRUM ANALYSER £99.95

**HI-FI OWNERS** and musicians — use Spectrum Analyser to create an exciting visual image of your music.

Watch your music displayed in bar chart format. Each bar corresponds to a particular range of frequencies ... **SEE** how tone controls, filters, etc. affect your music output.

**IDEAL** for enthusiasts as a visual aid to achieve the perfect bass, treble, balance, etc. Use as an imaginative and exciting addition to your existing music system.

3

NEW

## HOME £24.95
## ACCOUNTS

Fully GEM-based, very simple to use. This complete home accounting package will produce your own statements to check bank account(s)/charges, credit cards, etc. Full graphics facility and much, much more.

(0395) 45059

DIGITA INTERNATIONAL

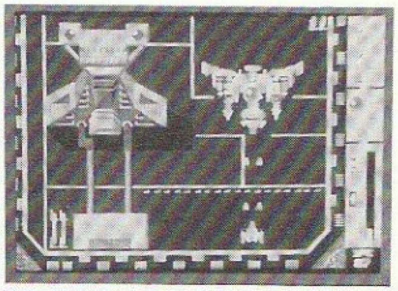
Kelsey House
Barns Road
Budleigh Salterton
East Devon
EX9 6HJ

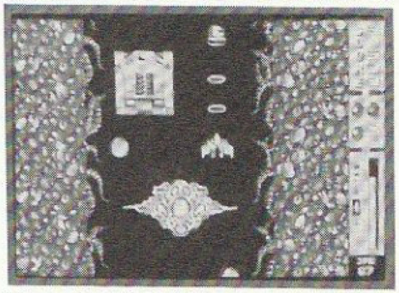
## Trauma

**From Infogrames**
**Price £19.95**
**Review by Tom Sharp**

Trauma is a no nonsense shoot 'em up in the tradition of Goldrunner and Hades Nebula, in fact it is so similar to Hades Nebula you could call it a rip off. The only thing is that it is more likely that Hades Nebula is a copy of Trauma than the other way round. Now that Nexus are no longer around I suppose we will never know! Regular readers of Monitor will have read the review of Hades Nebula and will know what to expect from Trauma (the only added feature I found was the Teleporting Pods which can jump your ship past some of the more difficult parts of a level).

For new readers and those that have not seen Hades Nebula, I'll go through the game concepts. You have a spaceship which flies across an alien backdrop and is attacked by enemy ships and fired on by stationary gun batteries, every time you are hit you lose energy, in addition there are flying objects to shoot that gain you points, they do not fire at you but do lose you vital energy if they touch you. On each level you have the chance to improve your ships fire power by making contact with five Cylindar Modules, but if you are hit after contact you lose the advantage. At the end of each of the four zones is a monster Hypership which holds the energy globe for that sector, if you can destroy the Hypership, the globe passes into your hands.





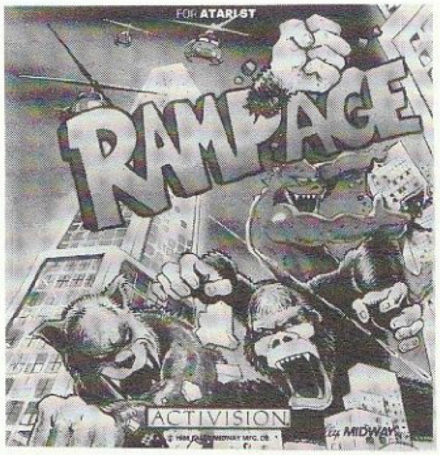
Basically that is the game. The graphics are good and so are the sound effects. Compared with Hades Nebula the game is slower and not so much fun, on the other hand as you don't die quite so easily, you do get a longer game play and a reasonable chance of getting to the finish.

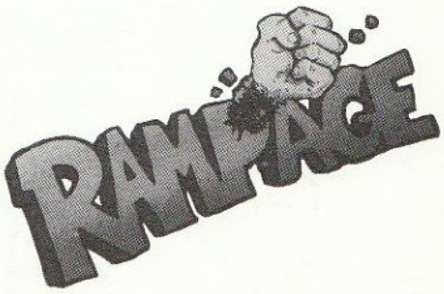
## Rampage

**From Activsion**
**Price £19.99**
**Review by Bill Dyer**

The story goes that one day at the Greaseburger Fast Food Emporium three unsuspecting customers got something even nastier than usual in their Big Muck's. Apparently the company's research division had accidentally shipped some of the experimental food additives, and the result (besides a foul aftertaste, nausea and indigestion) for the three luckless victims was an enhanced case of identity crisis. George, Lizzie and Ralph thought they were turning into 50 foot high monsters, a Gorilla, a Lizard and a Wolfman to be exact. Unfortunately for the inhabitants of the city, the three really have turned into rampaging monsters. They start to run amuck, smashing buildings, trampling on cars and eating people. The National Guard are called in and try to exterminate the menace using their helicopters and ground attack vehicles, but with little success.

Up to three can play the role of a monster (two on joysticks, one using the keyboard) or you can play alone, or you can take one monster and the computer controls the others. You climb up the sides of the buildings smashing the walls and windows with your fists searching for things to eat. People are tasty and so are the goldfish, the toast is nice but eating the toaster could be a shocking experience. Each creature's energy is shown on a bar at the top of the screen, and energy is drained away by eating something that disagrees with you or by getting hit by the guards. Of course, damaging a building drastically will cause it to collapse, make sure you jump clear before it falls or more energy will be lost.





Controlling the monster using the joystick is fairly straightforward, any movement of the joystick moves it in the desired direction. Pressing the fire button whilst the joystick is at rest (centred) causes the monster to jump. Pressing the fire button and moving the joystick will make the monster punch out or grab in the desired direction.

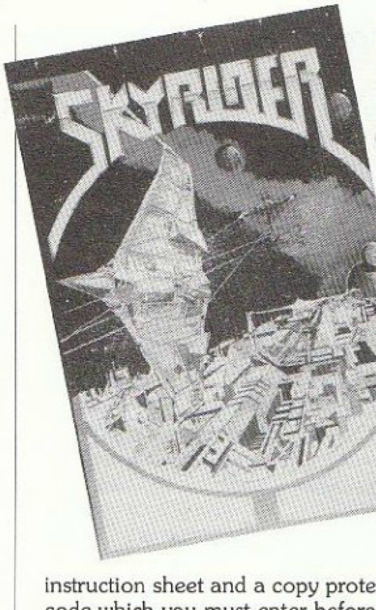
Rampage is an adaptation of a Bally Midway arcade game, and a very good implementation it is too! The graphics are very good with plenty of detail and there are 150 different screens to complete.

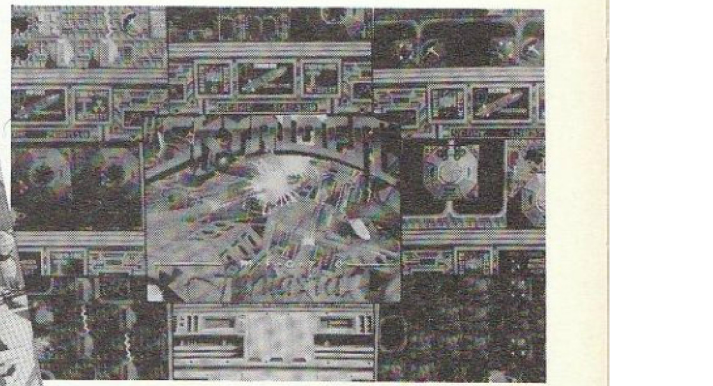
In general when an arcade classic is transposed to the smaller screen, a more or less guaranteed success is at hand. Activisions version of Rampage is no exception, it's sure to be a massive hit.

## Skyrider

From Creation
Price £14.95
Review by Stuart Rennie

Skyrider is a sideways scrolling arcade quality 'blast the enemy' game sporting excellent graphics and fast and furious game play. You fly your ship over the gigantic enemy vessel blasting any thing that moves and quite a lot that doesn't! Alien craft and force fields bar your way, and some things that you can't blast force you to manoeuvre through the strange landscape, a very tricky operation. But your deadliest foe is creeping radiation. You are only allowed a limited time to cover each sector before the radiation gets you. When you start the backdrop is cool reassuring blue, but it changes to green pretty soon and then to blood red, at this point you don't have much time left. You must destroy all deck targets and escape off the end of the sector or die!





Skyrider comes on two disks, an instruction sheet and a copy protection code which you must enter before you can play the game. Control of your Skyrider is by joystick and you can set control to normal or with inertia, this makes the ship much more difficult to handle but gives a more realistic effect. Use the 'I' key to turn inertia on or off. You can also turn off the music by the 'M' key, pause the game with 'P' and abort a game by pressing 'ESC'.

All in all, Skyrider is an excellent first product from a new company. I wish them success with it and any future new products. News is that Timelord is on its way, can't wait to see it.

# ST PROGRAMMING

**By Keith Mayhew Part Four**

Writing a program which makes full use of GEM AES's menus, windows, dialogues and icons is not as difficult as it may at first appear. This part of ST Programming is devoted to quite a large example program which will serve as a basis for discussing how to use and control multiple windows correctly in your programs.

The structure of a typical GEM application is 'event-driven'. This is a simple yet important concept, and is the primary key to understanding how the AES functions and how programs interact with it, so it will be described in detail before we study the example program.

## GEM Application Programs

The majority of programs which do not use GEM only perform very simple input or output to a text screen. Such programs have two basic properties: they execute in a loop waiting for input from the keyboard (for either a command line or a menu selection) and they are the only program which is executing, thus they have complete control over the use of the screen.

A program which uses GEM has added complications. Firstly it must be able to respond to multiple inputs: key presses, mouse movements and button clicks, window manipulations, menu selections, and so on. The user expects the application to be able to respond to any of these actions quickly and at any time.

Secondly, the program may have to support multiple output displays as the user opens new windows. It then has to update the windows as they need redrawing because either the program wants to display more information or has to rebuild part of an image which was previously obscured by another object.

Lastly, a GEM application is expected to be able to share control with desk accessories. It is surprising how many programs do not behave properly with accessories, overwriting their windows or, in the worst case, crashing the system!

## AES Events and Messages

To ease the burden on an application of having to constantly check all of its inputs, the AES provides an event manager. The structure of a GEM program then becomes a loop with a single call to the event manager informing it of which types of events it wishes to respond to should they occur.

After making an event call the program is suspended and, when a suitable event occurs, the AES restarts the program again. The program then has to decide what to do with the event or events which have occurred, which may include ignoring them altogether if it so wishes. If it responds to an event then it should do so by performing the appropriate actions and returning as soon as possible ready for the next event call. Of course, the time when the event call is not made again is when the program is ready to exit.

Although the above description is correct from the application's point of view, it is not the whole story of the event manager. As desk accessories can be resident as well as the main application it is necessary for control to be switched from one to another. It is the event manager which provides this control, giving the system a simple but effective form of multi-tasking. Until a program makes an event call, then the AES will not be able to give another program a chance to run, which is why it is important for event calls to be made as frequently as possible. This form of multi-tasking is called 'non-pre-emptive' as the AES has to wait for the application to make another event call before it can switch tasks, i.e. it cannot interrupt, or 'pre-empt', a program in mid-execution.

An advantage of the event calls is that optimum use can be made of the CPU's time. Rather than each program constantly checking its inputs, such as the mouse and keyboard, the event manager collects the requests together from all the programs in the system and performs the checks on inputs itself. It then decides which program to resume.

There is one very flexible event which the AES supports called a message. A message can be sent from any program to any other and consists of a sequence of bytes conveying some meaning. There are several predefined messages which can be sent by the AES to an application program to inform it of certain actions which it may need to take. There is however nothing to stop an application passing arbitrary messages to and from another as long as both understand the messages and that the originator of the first message knows the name of the other program.

There is an internal program in the AES, called the screen manager, which is responsible for system functions such as the manipulation of windows and the interactions with the menu bar at the top of the screen. It is the screen manager which sends the system messages to programs informing them of menu and window operations as they occur.

## AES Windows

Listing 1 is a C program which allows the user to open, close and manipulate windows. If you wish to compile this program you will require four 'header' files, which are referred to as 'defs.h', 'gem.h', 'events.h', and 'window.h' in Listing 1. This series will be supporting three of the most popular compilers currently available which are Metacomco's Lattice C, Mark Williams C and Megamax C. It was described in Part Two of this series that a set of type definitions help in porting programs between different compilers. Listings 2a, 2b and 2c provide the appropriate definitions for the current versions of each of the three compilers mentioned. These definitions should also work with older versions of the compilers but it would be wise to check them just in case. If you wish to use another make of compiler then you will have to construct a new set of definitions.

Which ever compiler you choose, the appropriate set of definitions should be in the file 'defs.h'. Listing 3 provides miscellaneous GEM definitions for the file 'gem.h'. Listing 4, file 'event.h', gives the definitions for the use of the event manager, while Listing 5, file 'window.h', gives the definitions for use with the window library.

To compile the program under Lattice C requires the use of the linker control file 'CGEM.LNK' which is supplied with it and the compiler flag '-n' for long label names. Mark Williams C requires the option '-VGEM' to 'cc'. Megamax C does not require any special options and will directly compile the program.

When the program is executed, it changes the mouse shape to an arrow, clears the menu bar area at the top of the screen and writes the centralised string 'WINDOWS DEMO' in it. When the mouse button is clicked on the grey 'desktop' background a new window is opened displaying a pattern in it. Up to four windows can be opened in this way, each displaying its own pattern. The windows can be re-sized, moved,

```
#include "defs.h"
#include "gem.h"
#include "event.h"
#include "window.h"

typedef struct WINDOW
{  WORD          w_handle;
   BOOLEAN       full;
   CHAR          *title_p;
   VOID          (*fn_p)();
   WORD          pattern;
} WINDOW;

VOID          events(), w_redraw(), w_top(), w_full();
VOID          w_size(), w_move(), w_close();
WINDOW        *find_window();
BOOLEAN       new_window();
VOID          fill_rect();
BOOLEAN       rect_intersect();
VOID          clip_rect(), rect_to_array();
VOID          rect_expand(), rect_shrink();
VOID          send_redraw(), w_get_rect(), w_set_rect();
WORD          phys_handle(), open_vwork();
VOID          fn1(), fn2(), fn3(), fn4();

#define  MAX_WINDOWS     4
#define  MAX_PATTERN     24
#define  W_MIN_W         (14 * _cell_w)
#define  W_MIN_H         (7 * _cell_h)

#ifndef  LATTICE_3_04
WORD          contrl[12];
WORD          intin[128], intout[128];
WORD          ptsin[128], ptsout[128];
#endif

#ifdef   MWC_2_0
   extern  WORD          gl_apid;
#endif

WORD          _process_id;
WORD          _num_windows;
RECT          _desk_rect;
RECT          _open_size;
WORD          _cell_w, _cell_h;

WINDOW   _windows[MAX_WINDOWS] =
{  { W_NO_HANDLE, FALSE, " Window 1 ", fn1, 1 },
   { W_NO_HANDLE, FALSE, " Window 2 ", fn2, 2 },
   { W_NO_HANDLE, FALSE, " Window 3 ", fn3, 3 },
   { W_NO_HANDLE, FALSE, " Window 4 ", fn4, 4 }
};

main()
{
   WORD          vdi_handle, dummy, attributes[10];
   RECT          menu_rect;

   _process_id = appl_init();
#ifdef   MWC_2_0
   _process_id = gl_apid;
#endif
   mouse_arrow();
```

```
   vdi_handle = open_vwork(phys_handle());
   vqt_attributes(vdi_handle, &attributes[0]);
   _cell_w = attributes[8];
   _cell_h = attributes[9];
   w_get_rect(W_DESK_HANDLE, WF_WORKXYWH, &_desk_rect);
   _open_size.x = _desk_rect.x + _desk_rect.w / 4;
   _open_size.y = _desk_rect.y + _desk_rect.h / 4;
   _open_size.w = _desk_rect.x + _desk_rect.w / 2;
   _open_size.h = _desk_rect.y + _desk_rect.h / 2;
   _num_windows = 0;
   w_get_rect(W_DESK_HANDLE, WF_CURRXYWH, &menu_rect);
   menu_rect.h -= _desk_rect.h + 1;
   fill_rect(vdi_handle, (WORD)0, (WORD)0, &menu_rect);
   vst_alignment(vdi_handle, (WORD)1, (WORD)5, &dummy, &dummy);
   mouse_off();
   v_gtext(vdi_handle, _desk_rect.w / 2, (WORD)0, "WINDOWS DEMO");
   mouse_on();
   events(vdi_handle);
   v_clsvwk(vdi_handle);
   appl_exit();
}

VOID          events(vdi_handle)
WORD          vdi_handle;
{
   WORD          event_flags, dummy, clicks, i;
   WORD          mouse_x, mouse_y, button;
   WORD          message[8];
   BOOLEAN       done;
   WINDOW        *window_p;
   RECT          rect;

   done = FALSE;
   while (!done)
   {  event_flags = evnt_multi(MU_BUTTON | MU_MESAG,
            (WORD)2, (WORD)1, (WORD)1,
            (WORD)0, (WORD)0, (WORD)0, (WORD)0, (WORD)0,
            (WORD)0, (WORD)0, (WORD)0, (WORD)0, (WORD)0,
            &message[0],
            (WORD)0, (WORD)0,
            &mouse_x, &mouse_y, &dummy, &dummy, &dummy, &clicks);
      wind_update(BEG_UPDATE);
      if ((event_flags & MU_BUTTON) != 0)
      {  if (clicks == 2)
         {  for (i = 0; i < MAX_WINDOWS; i++)
            {  if (_windows[i].w_handle != W_NO_HANDLE)
                  w_close(&_windows[i]);
            }
            done = TRUE;
         }
         else
         {  window_p = find_window(wind_find(mouse_x, mouse_y));
            if (window_p == (WINDOW *)NIL)
            {  do
               {  graf_mkstate(&mouse_x, &mouse_y, &button, &dummy);
               } while ((button & B_LEFT) != 0);
               window_p = find_window(wind_find(mouse_x, mouse_y));
               if (window_p == (WINDOW *)NIL)
               {  if (!new_window())
                  {  form_alert((WORD)1,
                        "[3][There are no more windows.][OK]");
                  }
               }
         }
         else
         {  window_p->pattern++;
            if (window_p->pattern > MAX_PATTERN)
               window_p->pattern = 1;
            w_get_rect(window_p->w_handle, WF_CURRXYWH, &rect);
            send_redraw(_process_id, window_p->w_handle, &rect);
         }
      }
   }
   if ((event_flags & MU_MESAG) != 0)
   {  window_p = find_window(message[3]);
      if (window_p != (WINDOW *)NIL)
      {  switch (message[0])
         {  case WM_REDRAW:
               w_redraw(vdi_handle, window_p, (RECT *)&message[4]);
               break;
            case WM_CLOSED:
               w_close(window_p);
               break;
            case WM_TOPPED:
               w_top(window_p);
               break;
            case WM_FULLED:
               w_full(window_p);
               break;
            case WM_SIZED:
               w_size(window_p, (RECT *)&message[4]);
               break;
            case WM_MOVED:
               w_move(window_p, (RECT *)&message[4]);
               break;
            }
         }
      }
      wind_update(END_UPDATE);
   }
}

VOID           w_redraw(vdi_handle, window_p, redraw_rect_p)
WORD           vdi_handle;
WINDOW         *window_p;
RECT           *redraw_rect_p;
{
   RECT           draw_rect;
   RECT           work_rect;

   w_get_rect(window_p->w_handle, WF_WORKXYWH, &work_rect);
   rect_intersect(&_desk_rect, redraw_rect_p);
   w_get_rect(window_p->w_handle, WF_FIRSTXYWH, &draw_rect);
   while (draw_rect.w > 0 && draw_rect.h > 0)
   {  if (rect_intersect(redraw_rect_p, &draw_rect))
      {  clip_rect(vdi_handle, &draw_rect);
         (*(window_p->fn_p))(vdi_handle, window_p->pattern, &work_rect);
      }
         w_get_rect(window_p->w_handle, WF_NEXTXYWH, &draw_rect);
   }
}

VOID           w_top(window_p)
WINDOW         *window_p;
{
   wind_set(window_p->w_handle, WF_TOP,
         (WORD)0, (WORD)0, (WORD)0, (WORD)0);
}

VOID           w_full(window_p)
WINDOW         *window_p;
{
   RECT           new_rect;

   if (window_p->full)
      w_get_rect(window_p->w_handle, WF_PREVXYWH, &new_rect);
   else
      w_get_rect(window_p->w_handle, WF_FULLXYWH, &new_rect);
   w_set_rect(window_p->w_handle, WF_CURRXYWH, &new_rect);
   window_p->full = !window_p->full;
}

VOID           w_size(window_p, rect_p)
WINDOW         *window_p;
RECT           *rect_p;
{
   if (rect_p->w < W_MIN_W)
      rect_p->w = W_MIN_W;
   if (rect_p->h < W_MIN_H)
      rect_p->h = W_MIN_H;
   w_set_rect(window_p->w_handle, WF_CURRXYWH, rect_p);
   window_p->full = FALSE;
}

VOID           w_move(window_p, rect_p)
WINDOW         *window_p;
RECT           *rect_p;
{
   if (rect_p->w < W_MIN_W)
      rect_p->w = W_MIN_W;
   if (rect_p->h < W_MIN_H)
      rect_p->h = W_MIN_H;
   w_set_rect(window_p->w_handle, WF_CURRXYWH, rect_p);
   window_p->full = FALSE;
   send_redraw(_process_id, window_p->w_handle, rect_p);
}

VOID           w_close(window_p)
WINDOW         *window_p;
{
   RECT           size;

   w_get_rect(window_p->w_handle, WF_CURRXYWH, &size);
   rect_shrink(&size);
   wind_close(window_p->w_handle);
   wind_delete(window_p->w_handle);
   window_p->w_handle = W_NO_HANDLE;
   _num_windows--;
}

WINDOW         *find_window(w_handle)
WORD           w_handle;
{
   WORD           i;
   WINDOW         *window_p;

   window_p = (WINDOW *)NIL;
   for (i = 0; i < MAX_WINDOWS; i++)
   { if (_windows[i].w_handle == w_handle)
      { window_p = &_windows[i];
         break;
      }
   }
   return (window_p);
}

BOOLEAN        new_window()
{
   WORD           w_handle, i;

   if (_num_windows == MAX_WINDOWS)
      return (FALSE);
   else
   { w_handle = wind_create(NAME | CLOSER | FULLER | MOVER | SIZER,
            _desk_rect.x, _desk_rect.y, _desk_rect.w, _desk_rect.h);
      for (i = 0; i < MAX_WINDOWS; i++)
      { if (_windows[i].w_handle == W_NO_HANDLE)
         { _windows[i].w_handle = w_handle;
            _windows[i].full = FALSE;
            break;
         }
      }
      wind_set(w_handle, WF_NAME, HW_ADDR(_windows[i].title_p),
            LW_ADDR(_windows[i].title_p), (WORD)0, (WORD)0);
      rect_expand(&_open_size);
      wind_open(w_handle,
            _open_size.x, _open_size.y, _open_size.w, _open_size.h);
      _num_windows++;
      return (TRUE);
   }
}

VOID           fill_rect(vdi_handle, style, pattern, rect_p)
WORD           vdi_handle, style, pattern;
RECT           *rect_p;
{
   WORD           array[4];

   vsf_interior(vdi_handle, style);
   vsf_style(vdi_handle, pattern);
   rect_to_array(rect_p, &array[0]);
   vr_recfl(vdi_handle, &array[0]);
}

BOOLEAN        rect_intersect(src_p, dst_p)
RECT           *src_p, *dst_p;
{
   WORD           x, y;

   x = max(src_p->x, dst_p->x);
   y = max(src_p->y, dst_p->y);
   dst_p->w = min(src_p->x + src_p->w, dst_p->x + dst_p->w) - x;
   dst_p->h = min(src_p->y + src_p->h, dst_p->y + dst_p->h) - y;
   dst_p->x = x;
   dst_p->y = y;
   return (dst_p->w > 0 && dst_p->h > 0);
}

VOID           clip_rect(vdi_handle, rect_p)
WORD           vdi_handle;
RECT           *rect_p;
```

```
{
   WORD           array[4];

   rect_to_array(rect_p, &array[0]);
   vs_clip(vdi_handle, (WORD)1, &array[0]);
}

VOID           rect_to_array(rect_p, array)
RECT           *rect_p;
WORD           array[];
{
   array[0] = rect_p->x;
   array[1] = rect_p->y;
   array[2] = rect_p->x + rect_p->w - 1;
   array[3] = rect_p->y + rect_p->h - 1;
}

VOID           rect_expand(rect_p)
RECT           *rect_p;
{
   graf_growbox(rect_p->x + (rect_p->w - W_MIN_W) / 2, W_MIN_W,
         rect_p->y + (rect_p->h - W_MIN_H) / 2, W_MIN_H,
         rect_p->x, rect_p->y, rect_p->w, rect_p->h);
}

VOID           rect_shrink(rect_p)
RECT           *rect_p;
{
   graf_shrinkbox(rect_p->x + (rect_p->w - W_MIN_W) / 2, W_MIN_W,
         rect_p->y + (rect_p->h - W_MIN_H) / 2, W_MIN_H,
         rect_p->x, rect_p->y, rect_p->w, rect_p->h);
}

VOID           send_redraw(process_id, w_handle, rect_p)
WORD           process_id;
WORD           w_handle;
RECT           *rect_p;
{
   static WORD         message[8];

   message[0] = WM_REDRAW;
   message[1] = process_id;
   message[2] = 0;
   message[3] = w_handle;
   message[4] = rect_p->x;
   message[5] = rect_p->y;
   message[6] = rect_p->w;
   message[7] = rect_p->h;
   appl_write(process_id, (WORD)16, &message[0]);
}

VOID           w_get_rect(w_handle, type, rect_p)
WORD           w_handle, type;
RECT           *rect_p;
{
   wind_get(w_handle, type,
         &rect_p->x, &rect_p->y, &rect_p->w, &rect_p->h);
}

VOID           w_set_rect(w_handle, type, rect_p)
WORD           w_handle, type;
RECT           *rect_p;
{
   wind_set(w_handle, type,
         rect_p->x, rect_p->y, rect_p->w, rect_p->h);
}

WORD          phys_handle()
{
   WORD          dummy;

   return (graf_handle(&dummy, &dummy, &dummy, &dummy));
}

WORD          open_vwork(phys_handle)
WORD          phys_handle;
{
   WORD          work_in[11], work_out[57];
   WORD          handle, i;

   for (i = 0; i <= 9; i++)
      work_in[i] = 1;
   work_in[10] = 2;
   handle = phys_handle;
   v_opnvwk(&work_in[0], &handle, &work_out[0]);
   return (handle);
}

VOID          fn1(vdi_handle, pattern, work_rect_p)
WORD          vdi_handle;
WORD          pattern;
RECT          *work_rect_p;
{
   mouse_off();
   fill_rect(vdi_handle, (WORD)2, pattern, work_rect_p);
```

```
   mouse_on();
}

VOID          fn2(vdi_handle, pattern, work_rect_p)
WORD          vdi_handle;
WORD          pattern;
RECT          *work_rect_p;
{
   mouse_off();
   fill_rect(vdi_handle, (WORD)2, pattern, work_rect_p);
   mouse_on();
}

VOID          fn3(vdi_handle, pattern, work_rect_p)
WORD          vdi_handle;
WORD          pattern;
RECT          *work_rect_p;
{
   mouse_off();
   fill_rect(vdi_handle, (WORD)2, pattern, work_rect_p);
   mouse_on();
}

VOID          fn4(vdi_handle, pattern, work_rect_p)
WORD          vdi_handle;
WORD          pattern;
RECT          *work_rect_p;
{
   mouse_off();
   fill_rect(vdi_handle, (WORD)2, pattern, work_rect_p);
   mouse_on();
}
```

Listing 1. Window demo program.

```
/*-------------------*/
/*  LATTICE C V3.04  */
/*-------------------*/

#define  LATTICE_3_04

typedef  char       CHAR;
typedef  char       BYTE;
typedef  short      WORD;
typedef  long       LONG;
typedef  short      BOOLEAN;

#define  VOID       void

#define  FALSE      0
#define  TRUE       1

#define  min(x, y)  ((x) <= (y) ? (x) : (y))
#define  max(x, y)  ((x) >= (y) ? (x) : (y))

#define  NIL        0
```

Listing 2a. Lattice C definitions file.

```
/*-----------------------*/
/*  MARK WILLIAMS C V2.0  */
/*-----------------------*/

#define  MWC_2_0

typedef  char       CHAR;
typedef  char       BYTE;
typedef  int        WORD;
typedef  long       LONG;
typedef  short      BOOLEAN;

#define  VOID       void

#define  FALSE      0
#define  TRUE       1

#define  min(x, y)  ((x) <= (y) ? (x) : (y))
#define  max(x, y)  ((x) >= (y) ? (x) : (y))

#define  NIL        0
```

Listing 2b. Mark Williams C definitions file.

```
/*------------------*/
/*  MEGAMAX C V1.1  */
/*------------------*/

#define  MMC_1_1

typedef  char        CHAR;
typedef  short       BYTE;
typedef  int         WORD;
typedef  long        LONG;
typedef  short       BOOLEAN;

#define  VOID

#define  FALSE       0
#define  TRUE        1

#define  min(x, y)   ((x) <= (y) ? (x) : (y))
#define  max(x, y)   ((x) >= (y) ? (x) : (y))

#define  NIL         0
```

Listing 2c. Megamax C definitions file.

```
/*-------------*/
/*  G E M . H  */
/*-------------*/

/* AES rectangle structure. */
typedef struct RECT
{
   WORD  x;
   WORD  y;
   WORD  w;
   WORD  h;
} RECT;

/* Low and high words of an address. */
#define  LW_ADDR(a)  (WORD)((LONG)(a)&0xFFFF)
#define  HW_ADDR(a)  (WORD)((LONG)(a)>>16)

/* Mouse button flags. */
#define  B_LEFT          (WORD)0x01
#define  B_RIGHT         (WORD)0x02

/* Keyboard 'shift', 'control' and 'alternate' flags. */
#define  K_RSHIFT        (WORD)0x01
#define  K_LSHIFT        (WORD)0x02
#define  K_CTRL          (WORD)0x04
#define  K_ALT           (WORD)0x08

/* Mouse control functions. */
#define  mouse_off()    graf_mouse((WORD)256, (LONG)0)
#define  mouse_on()     graf_mouse((WORD)257, (LONG)0)
#define  mouse_arrow()  graf_mouse((WORD)0, (LONG)0)
#define  mouse_busy()   graf_mouse((WORD)2, (LONG)0)
```

Listing 3. GEM.H file.

```
/*-----------------*/
/*  E V E N T . H  */
/*-----------------*/

/* Event 'multi' flags. */
#define  MU_KEYBD       (WORD)0x01
#define  MU_BUTTON      (WORD)0x02
#define  MU_M1          (WORD)0x04
#define  MU_M2          (WORD)0x08
#define  MU_MESAG       (WORD)0x10
#define  MU_TIMER       (WORD)0x20

/* Message event codes. */
#define  MN_SELECTED    10
#define  WM_REDRAW      20
#define  WM_TOPPED      21
#define  WM_CLOSED      22
#define  WM_FULLED      23
#define  WM_ARROWED     24
#define  WM_HSLID       25
#define  WM_VSLID       26
#define  WM_SIZED       27
#define  WM_MOVED       28
#define  AC_OPEN        40
#define  AC_CLOSE       41
```

Listing 4. EVENT.H file.

```
/*-------------------*/
/*  W I N D O W . H  */
/*-------------------*/

/* Window creation flags. */
#define  NAME            (WORD)0x0001
#define  CLOSER          (WORD)0x0002
#define  FULLER          (WORD)0x0004
#define  MOVER           (WORD)0x0008
#define  INFO            (WORD)0x0010
#define  SIZER           (WORD)0x0020
#define  UPARROW         (WORD)0x0040
#define  DNARROW         (WORD)0x0080
#define  VSLIDE          (WORD)0x0100
#define  LFARROW         (WORD)0x0200
#define  RTARROW         (WORD)0x0400
#define  HSLIDE          (WORD)0x0800

/* Window 'set/get' flags. */
#define  WF_KIND         (WORD)1
#define  WF_NAME         (WORD)2
#define  WF_INFO         (WORD)3
#define  WF_WORKXYWH     (WORD)4
#define  WF_CURRXYWH     (WORD)5
#define  WF_PREVXYWH     (WORD)6
#define  WF_FULLXYWH     (WORD)7
#define  WF_HSLIDE       (WORD)8
#define  WF_VSLIDE       (WORD)9
#define  WF_TOP          (WORD)10
#define  WF_FIRSTXYWH    (WORD)11
#define  WF_NEXTXYWH     (WORD)12
#define  WF_NEWDESK      (WORD)14
#define  WF_HSLSIZE      (WORD)15
#define  WF_VSLSIZE      (WORD)16
#define  WF_SCREEN       (WORD)17

/* Window 'calculate' flags. */
#define  WC_BORDER       (WORD)0
#define  WC_WORK         (WORD)1

/* Window and mouse 'update' control. */
#define  END_UPDATE      (WORD)0
#define  BEG_UPDATE      (WORD)1
#define  END_MCTRL       (WORD)2
#define  BEG_MCTRL       (WORD)3

/* Window handles. */
#define  W_NO_HANDLE     (WORD)(-1)
#define  W_DESK_HANDLE   (WORD)0
```

Listing 5. WINDOW.H file.

'fulled', 'topped', or closed again. If the mouse button is clicked within a window then its pattern is changed to the next one in a sequence of twenty-four patterns; if the button is kept pressed then the patterns will cycle rapidly. The program can only be exited by double-clicking the mouse button, either on the desktop or within a window. Figure 1 is a screen dump of the program's display with all four of its windows open.

Referring to Listing 1, the first thing done is the inclusion of the four header files. The definition which follows this is for a structure called 'WINDOW', later an array of four of these is declared, one for each of the windows which can be open. Each WINDOW structure contains an AES window handle, which is just a number identifying a window in the same way as a VDI workstation. As window handles start at zero, the handles are initialised to 'W_NO_HANDLE', which has a value of -1, to indicate that the windows are not open. We shall see the use of the other entries in the WINDOW structure later.

Next, all the programs functions are declared so that their return types are known. 'MAX_WINDOWS' determines the maximum number of windows allowed; although this is set to four, the maximum number of windows which can be open at any one time is seven. In fact, the desktop is a window itself, its handle number is zero and is defined as 'W_DESK_HANDLE' in 'window.h'. The desktop window is special though as it does not have any border components and you should not attempt to close it! 'W_MIN_W' and 'W_MIN_H' are the minimum width and height of a window, defined to be 14 character cell widths by 7 character cell heights.

The declaration of the five VDI arrays are not made if the compiler is Lattice because they are already defined in its library. Similarly, if the compiler is Mark Williams then the external variable 'gl_apid' is declared. This is because the process i.d. returned by 'appl_init()' is always -1 with that compiler and its actual value is returned via 'gl_apid', hence the extra assignment to '_process_id' in the 'main()' function if the Mark Williams compiler is being used. A value of -1 for the process i.d. will mean that you get no pattern change when clicking in a window, which is a good indication of whether your compiler deals with 'appl_init()' correctly or not.

After obtaining the process, or application i.d. the mouse is set to an arrow form and a virtual workstation is opened. The width and height of a character cell is determined by the VDI inquire function 'vqt_attributes()' and are stored in '_cell_w' and '_cell_h'. Thus, 'w_get_rect()' is then used to determine the desktop's window area. The code 'WF_WORKXYWH' causes the work area of a window to be returned, i.e. the window's display area. In the case of the desktop it returns the dimensions of the rectangle which excludes the menu bar area (just the grey part).

The AES specifies rectangle areas by the x and y position of their upper left corner and their width and height; note that this is different from how the VDI specifies a rectangle. The file gem.h

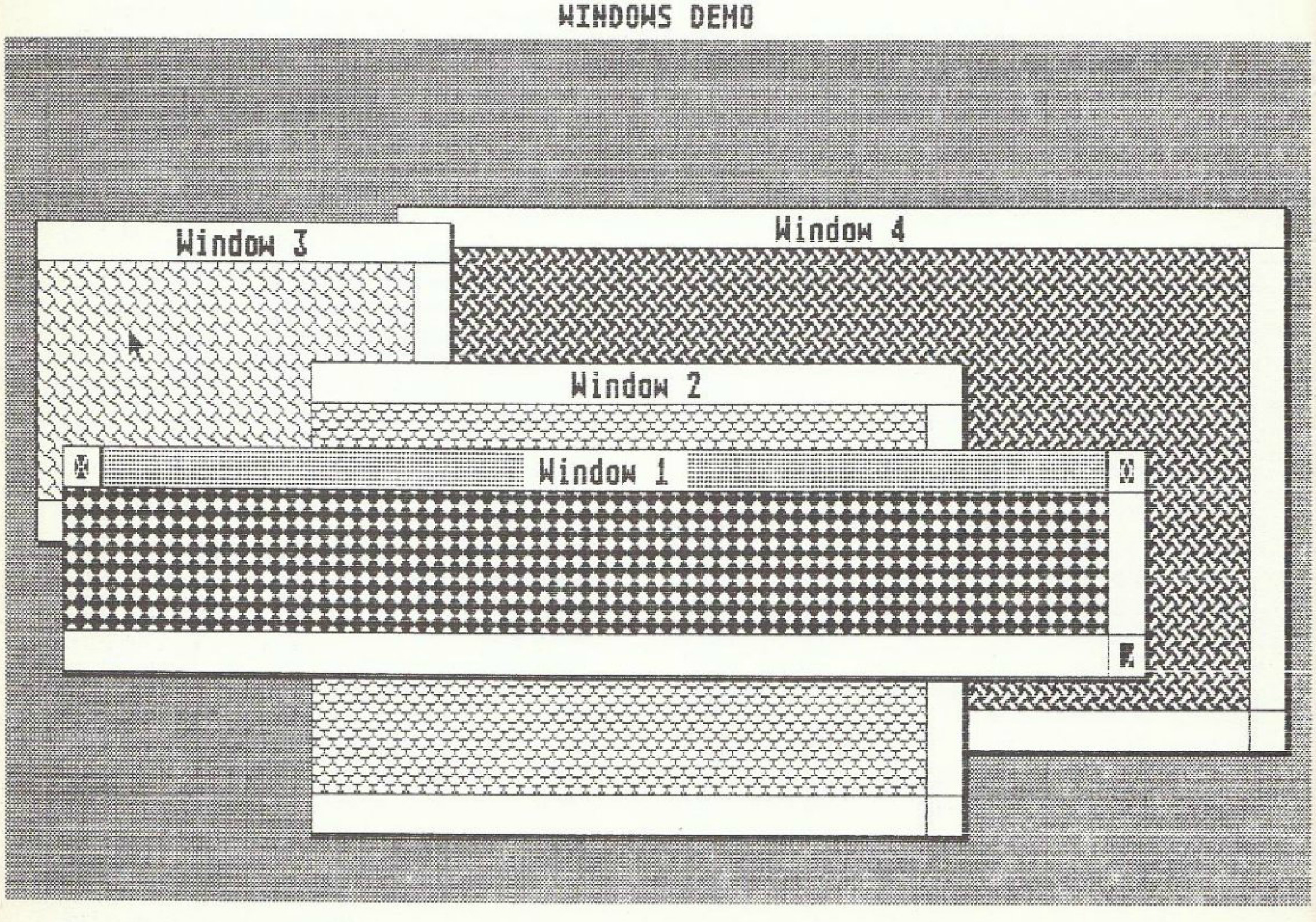


Figure 1. Screen dump of output.

defines a structure 'RECT' which we will use whenever we wish to specify an AES rectangle, hence '_desk_rect' is declared to be a RECT for the area of the desktop. '_open_rect' is set to be the size we want a window to be when it is first opened, done in relation to the size of the desktop. Note that by inquiring the size of the desktop, and using that as a basis for the open size of a window, we have avoided any dependence on screen resolution. Similarly, by asking for the character cell size as we did with 'vqt_attributes()', we obtained the size for the current resolution and the current font. Such methods should be used whenever possible to make your programs more flexible thus allowing them to work with both colour and monochrome systems, or even new ST systems.

Note that the function 'graf_handle()', which is used to get the VDI physical handle for the screen, also returns the character cell sizes but was not used in this program as its values for these are wrong!

Continuing with the function 'main()', the size of the entire desktop, including the menu bar, is obtained with a call to 'w_get_rect()' for 'WF_CURRXYWH'. With this value and the size of the desktop, the menu bar area can be calculated and cleared. The title string is then printed using 'v_gtext()' with centralised alignment. Lastly, 'events()' is called which will 'drive' the program.

The 'events()' function contains a 'while' loop which does not exit until the variable 'done' becomes TRUE. Each time round the loop an 'evnt_multi()' call is made which requests the events 'MU_BUTTON' and 'MU_MESSAG' for mouse button clicks and messages respectively. This means that it will never receive any other types of events. The next call of 'wind_update(BEG_UPDATE)' is important as it informs the AES that we might be about to alter a window in some way and so safely 'locks' all windows until the corresponding call of 'wind_update(END_UPDATE)' is made at the end of the event loop.

The main body of the event loop consists of a test for either a button event or a message event. It is important to note that the AES can send more than one event to the application and so both may be executed. This is why there is no 'else' before the second 'if'!

For a button event, the number of button clicks is tested. If it is two then any entries in '_windows' with a valid handle number have their associated window closed. The 'done' variable is then set to TRUE so that the program will exit.

For one click, 'wind_find()' is used with the mouse co-ordinates to get the handle of the window visible at that point. If the mouse is over the desktop then zero is returned. The function 'find_window' then searches the array '_windows' and returns a WINDOW pointer.

If this was NIL then the desktop was clicked on and so an attempt is made to open another window. Before doing this it waits for the user to let go of the mouse button, it then re-checks to see if the mouse is still over the desktop and only opens a window if it is. Should the 'new_window()' call return FALSE then an alert is displayed indicating the fact that there are no more windows to open.

Should the click have occurred within a window then the pattern number stored in the WINDOW structure is changed. Rather than redrawing the whole window pattern directly, a 'redraw' message event is sent to the application identified by '_process_id', which is of course itself, for the whole area of that window. Thus next time the event call is made it will receive a redraw for that window. Either method can be used.

There is a subtle point to raise at this stage about the way the screen manager monitors the mouse button. When the button is clicked over any part of an inactive window, i.e. not the top most one, then its owner's program is sent a message for it to be 'topped'. If the click occurred in the border of the active window then the screen manager waits for the mouse button to come up again at which point the program may be sent a message for re-sizing, moving, etc. Otherwise should the mouse be clicked on the desktop or the active window's work area then the owner of the active window is sent the button event, thus if an accessory happened to be the active window the main application would not receive button events. Similarly, keyboard events are only sent to the owner of the active window. The subtle point for the mouse button though, is that if it is clicked over the desktop or work area of the active window then the application will receive its button event but on returning to the screen manager, if the button is still down, that same application will continue to get the mouse events even if the mouse goes over an inactive window.

This effect manifests itself in this demo such that if you depress the mouse button over the top most window to change its pattern and keep the button pressed, the pattern will rapidly cycle as it continues to receive button events. If, however, the mouse is then moved onto an underlying window without touching the desktop then the window will not cause a 'topped' message. Instead, the next 'wind_find()' call will return the handle of the inactive window which might be partly obscured. It is therefore important that drawing is only done within the visible parts of that window. The moral of this story is that you should not make assumptions about which window is active!

The message events sent by the screen manager which the program responds to are as follows: redraw, closed, topped, fulled, sized and moved. Note that if, say, a 'sized' message is ignored then the user will see the animated box the screen manager produces for re-sizing but will not see the window actually change size - which has to be done by the application. This is an advantage of GEM as a window's actions can be either modified or completely ignored, as the application desires.

If the window handle specified with a message is valid then the appropriate routine is executed. The 'w_redraw()' function is the most complicated of these. A redraw message is sent to an application whenever part of one of its windows needs redrawing. The whole window could be redrawn but AES provides the size of the rectangle which is 'dirty' so only that area needs to be redrawn. This redraw area should first be intersected with the desktop area to ensure it is all visible. The 'rect_intersect()' function takes two rectangles and returns TRUE if they intersect, i.e. overlap, and if so updates the second rectangle to the size of the intersecting area.

The AES maintains a list of the rectangles which make up any particular window. If it is the active window then there is only one rectangle. However, for the other windows there can be an arbitrary number of rectangles to describe its visible regions. In Figure 1, 'window 2' has two rectangles in its list, 'window 3' has three and 'window 4' has four. Redrawing a window consists of getting the first rectangle in its list using 'WF_FIRSTXYWH' and, if it has height and width, intersecting it with the redraw area, clipping to the resulting area and redrawing that window. The rest of the rectangles are obtained with repeated calls for 'WF_NEXTXYWH'. In the program, the redrawing function for a window is obtained by a pointer field 'fn_p' in its associated WINDOW entry. These point to the routines 'fn1()' through 'fn4()' which draw the specified patterns. 'w_top()' is a simple function which just requests the AES to make the specified window the active one, after a user tries 'topping' it. 'w_full()' is called when the box in the upper right corner of a window is clicked on. The flag 'full' in the WINDOW structure indicates the current state of the window and the routine toggles this and changes the window's size either to its maximum or to its previous size using 'WF_FULLXYWH' and 'WF_PREVXYWH' respectively. And 'w_size()' and 'w_move()' are similar in that they alter the size and position of a window. Both of these functions can cause a window to be redrawn. For 'size' a window only generates a redraw message if it is being made larger, while 'move' generates one when a window is being moved from a position partly off the screen to one further on the screen. It is sometimes necessary for an application to always redraw a window on re-sizing in which case it is best to send a self-redraw because, if a window is becoming larger, AES is clever enough to spot the two redraw messages and merge them, thus removing the possibility of a double redraw.

When moving a window, the contents are usually 'blitted' thus avoiding a redraw. However, in this program, as a pattern is drawn it is necessary to redraw after a move and so a self-redraw is sent. The reason for this is that patterns are aligned to certain boundaries and so are unaligned if blitted by, say, one pixel. This causes problems if part of the pattern is then redrawn as it will be properly aligned and thus disjointed with the rest of the pattern! There are another two solutions to this kind of problem. The first is to align the position of the window whenever it is moved to, say, a word boundary in memory thus always keeping the patterns aligned correctly. The other, harder, solution is to draw the pattern in a special buffer and blit the buffer to the required window position.

The size and move functions also test to ensure that the specified area of the window is not less than 'W_MIN_W' and also 'W_MIN_H'. The reason for this is that if a window has no slider or arrow parts then it may be sized to a degenerate window which is totally unusable!!! 'new_window()' creates and opens a window. A call to 'wind_create()' specifies the components the window should have. Note that for a window to be movable, the flag 'MOVER' has to be set; the flag 'NAME' merely indicates a window is to have a name in the 'mover bar' area at its top. The return value from 'wind_create()' should be checked in most applications as accessories may restrict the number of available windows. In this program though, an application cannot be directly invoked as there is no menu bar. 'wind_crate()' also specifies the maximum, or full, size of a window.

Before the window is actually opened its name string is set with 'WF_NAME'. Note that the functions 'HW_ADDR()' and also 'LW_ADDR()' defined in 'gem.h' are required to break the address of the title string into a high and low word respectively. It is NOT portable to give just the pointer as a single argument in place of the two WORD arguments needed as is sometimes done!!! An expanding rectangle is drawn using 'rect_expand()' and finally the window is opened with 'wind_open()' at its default size specified by '_open_size'. Note that no draw function is called after a window is opened as a redraw message is automatically sent to the program by the AES.

After closing a window with 'w_close()' the 'wind_close()' function of the AES causes another window to be sent the 'topped' message. The window which will get this message is in fact the most recently used window before the one which was closed.

## Next Time

Having introduced a simple program, and looked at many of the more subtle parts of windows and events, you should now have a 'feel' for GEM programming. By trying your own small programs you should quickly learn how to use many of the relevant AES functions. If you have not got a good reference book for GEM then I can recommend 'Programmer's Guide to GEM' by Balma and Fitler, from Sybex. It provides a very readable guide to all the AES functions and the vast majority of the VDI functions.

Next time we will continue our study of the AES and will hopefully start covering object trees and resource files. I would also like some feedback from you, dear readers, on whether you would like to see a library of useful functions developed to simplify the task of AES programming.

# ST LIBRARY

**Librarian: Mike Stringer**

## Introduction

Allow me to tell you how the ST Library is going to be structured. Listed here are the disks currently available. I am expecting about thirty disks from North America, plus another dozen or so from some members over here. Still, we will be starting with a fair foundation upon which to build a very useful and valuable service to you, our readers.

The disks that I will be sending out are DS/DD but will be formatted for single sided use. Where the program requires 1 Meg formatting, these disks will be clearly marked and no additional fee will be requested. In other words, the fee will be the same, irrespective of the size of the program(s).

In some instances the files may be compressed. The necessary Archiving program will always be included on the disk, including the necessary info to allow you to convert them back to normal. In this way I will be able to put up to the equivalent of 500K of files on one, half-meg, disk.

In addition to the files, I will also include, if space permits, an up to date list of the library. The reason behind this is to keep you up to date at all times, you will not have to wait the three months, or so, for Monitor to arrive.

Because I have had very little response from you on how you want the Library to be structured, I have arranged it in the manner that seems the most logical and workable for me to provide a quick response to your requests.

Each disk will be filed under a heading according to the subject which the program/files relate. For example: LP1 is a Language disk, the subject is Pascal and it is the first in this particular section. Or, MMS1 is a MIDI disk containing files for Music Studio, again number 1.

There will also be a Support section which is intended to be used with programs/files for use with existing Commercial Software. For example, templates for VIP, Fonts for word processors or Printer Configurations and so on.

MIDI support files will be contained within the MIDI section because of the nature of the subject. I have given one example, but others already include Casio CZ Voices, 36 banks of voices for the Yamaha DX7 with the DXDROID, etc.

As other sections become available they will be introduced. Wherever possible, programs and files will be segregated to maintain integrity. If there is a demand for a mixture, I will try to oblige, this will be the exception, not the rule.

## What to do

The club has laid out a great deal of money to get the Library off the ground and in order to recoup these costs and to obtain new material, it is necessary to make a small charge. There are two services currently available. The first, you provide the disk with your request and the fee is £3.50. The second, we provide the disk (DS/DD) when the fee is £5.50. This includes all necessary return postage and packing.

Any member who submits material will have his disk returned, the contents having been copied into the Library, to be replaced by something very useful (or a request of your own) as a form of thanks. Please remember that if you do submit any material, it must qualify for the description of Public Domain, or something similar, i.e. no ripped off Commercial Software will be tolerated.

If at any time you wish to obtain the latest complete library list, just send a disk and £1.00, or just send £2.50 and we will supply a disk with the list recorded onto it.

## The ST Library is for subscribers only.

### SPECIAL THANKS

Season's greetings to you all and I would like to take this opportunity to send you all the very best wishes for 1988. Thank you all for the support you have shown to our new venture with the ST LIBRARY and I would like to thank especially those readers who have submitted programs for insertion into the library.

I think a special greeting should also be directed to those software houses who have contributed to the library with special programs and associated files. I am thinking, in particular, of Softbits (for Animatic), the lads at Eidersoft, 2 Bit Software (for ST Replay) and last, but not least, Laurence Wilkes. I would like to wish you all the greatest success with your products in 1988 and that you will be able to keep up the standards you established in 1987 and continue to produce so many interesting programs for us in the future.

I hope that those contributors who requested library programs over the Christmas period have at last received them. When I sent them out, I was unaware that local postal disputes were bubbling up and consequently, some may have been delayed/mislaid. If you were one of these, please drop me a line and I will rectify this immediately.

For your edification, I have been able to include another batch of new programs into our stock. These include some very good ART packages, animation, language, midi and sound samples.

For a complete listing of all the programs and files in the library please write in for a copy of the disk based library list. This file is now in excess of a 1/4 megabyte and the program TEMPUSDEMO is included to assist you.

Listed here are this quarters new additions.

### ADEMO 3

Art demo containing BIRDS, JUMPBALL, KISS and SINGSONG. 1/2 MB colour.

### ADEMO 4

Numerous examples of the graphic potential of the ST. 1 MB mono and colour.

### ADEMO 5

Three outstanding programs from TEX, which demonstrate just what is possible with the ST - colours (520 on screen at once!), great graphics and sound. 1/2 MB colour.

### ANIM 1

The 'SPINNING DOLLS' - takes a while to load, but the wait is worth while. A FANTASTIC demo. 1/2 MB colour.

### ANIM 2

The ROBOT from SAM. 1/2 MB.

### ART 10

An AUTO display of digitised photographs. 1/2 MB colour.

### ART 11

AUTO display of some naughty pictures, X rated. 1 MB colour.

### ART 12

Another X rated disk. 1/2 MB colour.

### ART 13

A series of digitised photographs with SPACE SHUTTLE and STAR WARS images. Brilliant. 1/2 MB colour.

### COMDEMO 2

CAD demo, cyberscape. Most impressive. 1 MB colour.

### GAMESUTIL 1

ARKENOID construction set. Allows you to construct and play your own screens. Lots of new screens included. 1 MB colour.

### LC 1

The .C and .H files to accompany the article in this issue written by Keith Mayhew. Because this is part of the magazine structure, the price for this type of library disk (and others still in the preparation stage) will be a nominal £2.50. 1/2 MB.

### LGFA 1

ROLL AND NUDGE program and source files written in GFA Basic by member Mike King. 1/2 MB colour.

### MDX Vol 1

55 new banks of 32 voices for the Yamaha DX7, for use with MDXORG. Supplied by Laurence Wilkes. More to follow!! 1/2 MB.

### MSAM 1

36 pop tunes with pictures. Music Studio compatible - MIDI output. 1/2 MB colour.

### SOUND 7

Sound sampled 'MATT'S MOOD'. 1 MB mono/colour.

### SOUND 8

Sound sampled 'SECRET SEPARATION'. 1 MB.

### SOUND 9

Sound sampled 'DIRE STRAIGHTS'. 1 MB.

### SOUND 10

Sound sampled 'MONEY FOR NOTHING'. 1 MB.

### SOUND 11

Pop tunes using sampled voices played in a very interesting way which results in an extremely long demo. From the ST REPLAY LADS! 1 MB colour.

### VIPTEM 4

Another batch of VIP templates - many with MACROs for use in home/small business. 20 templates that include spreedsheets for LEDGERS, PAYROLL, TAXATION, COOKBOOK, etc. 1/2 MB.

Requests should be sent to Mike Stringer, P.O. Box 3, Rayleigh, Essex, SS6 8LR.

# The Waddington MIDI Sequencer

Review by Michael Stringer

For the benefit of the uninitiated reader, a MIDI SEQUENCER is a computer tape-recorder. This particular recorder is not your run-of-the-mill, stereo recorder, but a mammoth 32 track device. In hardware terms, a recorder, such as this would cost anything between 50,000 and 150,000 pounds.

The next question, I hear you ask, is "Why do you need so many tracks?". The answer is directly connected to your individual requirements and also to your musical prowess. It is very much like owning a Porsche. It is capable of 150 m.p.h. or more, way above the speed limit. It is very satisfying to know that all that power is there, even if you will never use it. Or, it can be compared to a computer and the amount of RAM available. You may never use all the RAM that is available to you, but there are many who do use it all, and demand more. Hence the availability of 2 and 4 Meg Atari STs! The same thing applies to music and recorders. Not only is there a demand for vast amounts of RAM but also multi-channel recorders for orchestration.

MIDI has now been around for a few years and has inspired a new home recording industry. In the music industry, the end product is a two channel device, be it a disk, cassette, tape or compact - but the stages in their manufacture are very sophisticated and costly.

With this piece of software, a musician is able to produce, either for his/her own pleasure, or a quality demo for commercial appraisal, a most professional product. Ideally, a conventional multi-track recorder, reel to reel or cassette, would be very useful. All orchestrations could be compiled on the computer and then they in turn could be layered and mixed down into a traditional and convenient listening device.

The sequencer is a REAL TIME recorder. In other words, no matter what you play on the MIDI instrument, how you play it, or any pitch bending, voice changing and after touch are faithfully recorded to an accuracy of 1/100th of a beat.

The same thing applies to the proverbial cock up. This musical technical term signifies that a mistake, either musical or rhythm, has taken place. No matter, carry on until the end of the piece. The passage is re-played and when the cock up appears it is simply punched out. Using the same technique in reverse, a new passage can be punched in.

This feature is also available on a standard recorder, but there are features on this sequencer that are not usually found. Suppose there is a particularly difficult passage to negotiate. One simply slows the recording tempo down to a point where it is playable, record and playback at the correct speed and it is done. If this were attempted on a standard recorder, the passage would sound wrong, due to the increase of the pitch. Or, if there were inaccuracies in tempo, it is possible to QUANTISE the passage which straightens out any imbalances.

On loading the program, the first thing to notice is that it is GEM based. There are only four drop-down menus, but they contain some very useful features.

FILE is the standard storage/retrieval disk file device. It has been kept very simple - LOAD, SAVE and QUIT.

EDIT is concerned with MOVING, COPYING and DELETION of ALL or SELECTED tracks.

TRACKS allows you to edit the data contained within ALL or SELECTED tracks. QUANTISE has already been described. TRANSPOSE is a very useful feature. This allows you to raise or lower the notation throughout the range permitted by MIDI. It may be a simple change from a C in octave number 6 to an E in the same octave or it may be more dramatic - C6 to C2. In other words, we have lowered the whole keyboard by four octaves!

VELOCITY. If your MIDI instrument is capable of responding to touch sensitivity, this feature allows overall control on ALL tracks, selected REGION and selected TRACKS of the MAXIMUM or MINIMUM velocity. It is expressed as a percentage.

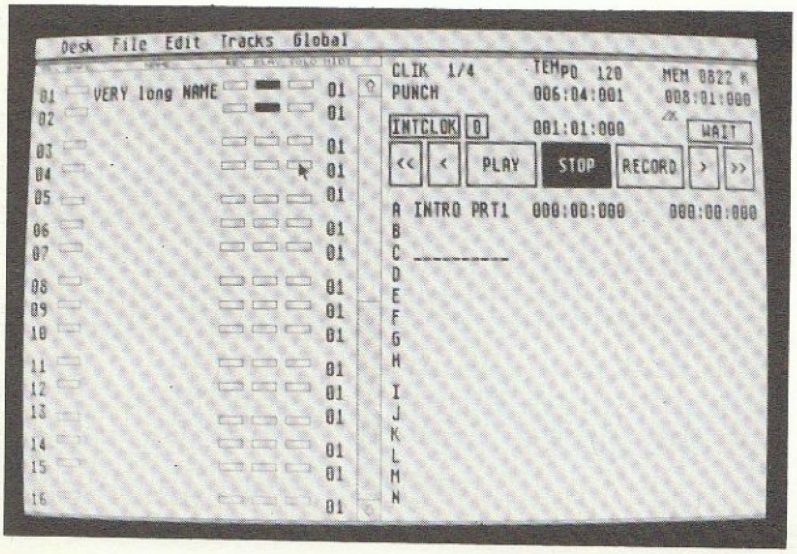
Also to be found in this menu are the PUNCH, LOOP and ERASE TRACK selections.

The fourth menu is GLOBAL. This contains a MIDI ALL-NOTE-OFF function, SAFETY, TIME SIGNATURE and MIDI. The selection of SAFETY invokes a dialogue box where you can toggle warning flags to confirm INSERT, DELETE, COPY and ERASE.

TIME SIGNATURE allows you to select, from a range the T/S for the piece. MIDI allows you to select AFTER TOUCH, MIDI CLOCK SEND, LOCAL CONTROL and MIDI THROUGH.

Here I will conduct you through the numerous screen features and icons. The left hand side of the screen deals with Track and Channel selection, Track Title info, three little boxes for RECORD, PLAY and SOLO, MIDI CHANNEL NUMBER SELECT for transmit or receive. There are sixteen channels visible, the remaining sixteen are accessed by clicking on the shaded bar down the centre. Some of the headings to the boxes are very minute, you may have to look closely to recognise them. The right hand of the screen deals with the recorder controls.

CLIK. This feature controls the beat of the METRONOME through the ST's internal speaker. The beats selectable are



1/4, normal, dotted and triplet; similarly notated for 1/8th and 1/16ths. To lower, position the mouse over the CL of CLIK, to raise, position over the IK.

TEMPO is the overall speed of record/playback. It is selectable over the range 40 to 215. The control of this function is similar to CLIK. To lower the rate click the mouse over TE and to raise, click over MPO.

In the top right hand corner is a memory remaining notice. This feature is invaluable when creating large compositions. Mine defaults to about 820K bytes and, depending on my use of Pitch Bending and After Touch, represents a lot of notes!! I mention these two synthesiser features because they really gobble up RAM.

The next feature is the PUNCH clock indicators. The start and finish meters are triggered by the mouse in the appropriate places.

The Metronome is a toggle icon, shaded is ON, unshaded is OFF. Turn the volume up on the ST for this feature.

Beneath the Punch indicator is the MIDI clock feature, which toggles between Internal and External clocks. The (0) box is the main tape position indicator to the right, zeroing, or reset button.

The main recorder's meter is neatly centred. As you can see, it is composed of three separate boxes, these correspond to BAR, BEAT and 1/100 of a BEAT.

Beneath these are the main recorder icons. Left double chevron is FAST REWIND, the single is SLOW rewind. The PLAY, STOP and RECORD boxes are self explanatory. The chevrons to the right are SLOW and FAST forward wind.

Beneath these buttons is a very useful work area where one can make notes on the various channel information. I use it mainly to make notes of VOICE information. Because I have many thousands of voices available for my synthesiser I use this to remind me of the name of the voice, which disk it is stored in and which bank of sounds in that disk where it can be found.

That completes the tour. Now let us see just how easy it is to use. First of all, connect the MIDI OUT of the ST to the MIDI IN of your MIDI instrument - be it synthesiser, drum machine, or whatever. Take the MIDI OUT lead from your synthesiser and connect the other end to the MIDI IN of the ST, switch it on and away we go!

Whenever I start a new piece I always ERASE ALL TRACKS from the TRACKS menu. Select the Metronome if required and also set the TEMPO to a rate that is appropriate.

Next, click on the little RECORD box on the left hand side of the screen, it is close to the NAME box adjacent to TRACK 1.

Select the RECORD button on the main control panel and start playing. You can also experiment with voice changes, pitch bending and after touch if you have these facilities on your instrument. Press the STOP button when you have finished.

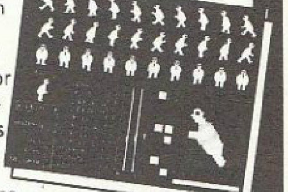
If the reset button (0) is now pressed, the tape counter zeros and the piece can be played back when you press the PLAY button.

If you would like to try something new, don't forget to clear the buffer from the TRACKS menu! If you intend to save a piece to disk, it will pay you to re-press the voice button on the synthesiser in order to voice-ident the beginning of the piece. Once you have familiarised yourself with the basic features you can start to experiment with looping and punching!

I am very fond of this piece of software. It compares very favourably with other sequencers on the market, some of which cost hundreds of pounds. It is very easy to use and also reliable.

Now to the nitty gritty. Availability. This program is from the ST LIBRARY, it is MSEQ1!!!

If you would like a copy of this program, see the ST Library page for details on how to send for programs.



## BACK ISSUES

Previous issues of Monitor are obtainable from the club for £1 plus 30p postage each. They contain many interesting and informative articles, hints and tips, program listings, reviews and practical advice. If you have missed out send for your copies of back issues today!! Please note that issues 1,2,3,4,5,6,7 & 9 are already sold out.

### Number 8.

Includes: Cracking the Code. 2 new series; Opening Out and Starting from Basics. Horizontal and vertical scrolling. Mask of the Sun, Sorcerer, Conan, Alley Cat, Ghostbusters and Spy vs Spy all reviewed. Programs include Quickplot, Nightmare Reflections and Matchbox.

### Number 10.

Includes: All about digitised pictures. How disk files work. Cracking the Code, Starting from Basics and What's MIDI all continue. Programs include: Disk Jacket, PCB Paranoia and 3D Maze. American Road Race, Kennedy Approach, Asylum, Red Moon and Wishbringer reviewed.

### Number 11.

Includes: RAM Talker for 400/800. Book reviews. MIDI programs. ST Hi-res Hat program. Hexadecimal Code generator. Reviews of Atariwriter Plus, Sidewinder, Koronis Rift, Electraglide, Mercenary, Fighter Pilot, Goonies and Alternate Reality. Plus Starting from Basics and Cracking the Code.

### Number 12.

Includes: Add-on circuits for various motors. Disk file handling. Matrices and Arrays explained. Write your own adventure. Space Invaders program. Reviews of Technicolour Dream, Eidolon and Action Biker. ST reviews include DB Master One, Time Bandit and Menu Plus.

### Number 13.

Includes: Omnimon and Ultimon compared. Data compression. Megamax C and Lattice C evaluated. Temper the sound of your 8 bit. Players and missiles explained. Programs include Graphics 8 page flipper, Demon adventure game. Reviews of Super 3D Plotter II, Planetarium, Price of Magik, Last V8 and Nuclear Nick. ST reviews include Cornerman, Cards and Major Motion.

### Number 14.

Includes: Display Lists. Adventurers sentence analyser. In depth look at Happy Revision 7. Graphics Modes. Video digitiser mods for use with XL/XE machines. Deathzone, a superb arcade game. Reviews of Crystal Raider, Molecule Man, Domain of the Undead, Laser Hawk, Rick Hanson, Colleen Music Compendium and Spellbreaker. ST reviews include Music Studio, Starglider, TrimBase, Electronic Pool, Easy Record and Pinball Factory.

### Number 15.

Includes: Player/missile priorities and interrupts. Turbo Basic commands and functions. 1050 write switch project. Enter commands directly in Basic. Whist card game for you to type in. DOS modifications. OS Controller Card evaluated. Reviews of Spitfire 40, Crumble's Crisis, Robot Knights and Replay. Intro to ST programming. ST Blitter. Reviews of Hollywood Hijinx, BCPL, K-Resource, Make, Micro-time Clock Card, Alternative, Trivia Challenge and Fast Basic.

### Number 16.

Includes: Character mapped modes and an introduction to scrolling. Using PLOT and DRAWTO in Graphics Zero. A useful hexadecimal converter program. Minotaur, a machine code monitor from Basic. Split screen effects for adventure writers. XIO for beginners. Mini Office II, Autoduel, Death Race, Sprong and Space Lobsters reviewed. ST section includes: How to use GEM with examples in C. Useful routines written in assembler. Six ST books reviewed. Hades Nebula, Airball, ST Replay, ST Digidrum, Crafton & Xunk, Animatic, Zoomracks II, Mousetrap, Prohibition and Barbarian are all reviewed.

### Number 17.

Includes: Vertical and horizontal scrolling routines. Berg, a super adventure set in the freezing waters of the north atlantic. Scrabble Crossword, a type in board game. A colour chart to adjust your TV with. Druid, Pirates of the Barbary Coast, The Dungeon and Lightspeed C reviewed. ST section includes: More useful routines in assembler, including a Degas picture display utility. GEM function calls such as VDI, AES, attribute, control, output and input. Terrorpods, GFA Draft, Fast ASM, M-Cache, Tempus and STuff reviewed.

## MONITOR ON DISK

All the 8 bit programs published in each issue of Monitor are now available pre-recorded on disk for you. No more need to spend frustrating hours of typing only to find the program won't run, and you are faced with the daunting task of bug hunting. The price is just £4.95 which includes postage and packing. Send a cheque/postal order made payable to the 'U.K. Atari Computer Owners Club' to Monitor Magazine, P.O. Box 3, Rayleigh, Essex SS6 8LR. If you live in Europe add 50p, if outside Europe add £1.00. Please allow 14 days for delivery. The following are available.

Monitor Disk 8.
Monitor Disk 9.
Monitor Disk 10.
Monitor Disk 11.
Monitor Disk 12.
Monitor Disk 13.
Monitor Disk 14.
Monitor Disk 15.
Monitor Disk 16.
Monitor Disk 17.

### Monitor Disk 18.

Includes: Basic checker, an error message program. RAMdisk mover, read disk contents quickly. Program to copy files using CIO. Bonus program: AQ an exciting adventure trading game.

# SUBSCRIPTION FORM

If you are not already a subscriber fill in this form (or a photocopy) and send it to the address below together with a cheque/postal order made payable to the 'U.K. Atari Computer Owners Club'. Your subscription entitles you to receive the next four issues of Monitor and enrols you as a member of the club. Please state from which issue number your subscription should commence. Annual subscription rates are £5.00 in the U.K. and Eire, £8.00 in Europe and surface delivery outside Europe, £12.00 Airmail delivery outside Europe.

**Don't delay do it today!!**

I want to enrol as a club member and receive Monitor magazine. I enclose a cheque/postal order for £5.00/£8.00/£12.00. Please send me issue ..........

Name ____________________

Address ____________________

Post Code ____________________

My system is an: XL/XE □ ST □ XL/XE and ST □

**Send to Monitor, P.O. Box 3, Rayleigh, Essex SS6 8LR.**

# NEW LOW PRICE ST!

**£260**

**+VAT=£299**

## ONLY FROM SILICA

Finally, there's a personal computer that not only solves problems like other computers, but also solves the one problem that other computers have created. Affordability. Silica Shop are pleased to present the ST range of personal/business computers from Atari. The ST was designed utilizing the most recent breakthroughs in semiconductor technology, producing a personal computer that performs tasks with fewer parts. Which means it costs less to make. And less to buy. The latest ST computers now include built in power supplies and built in disk drives. The TOS operating system and GEM window environment are now on ROM chips which are already installed in the ST keyboard. This enables automatic instant booting when you switch on. Silica Shop are pleased to offer the complete Atari ST range. Our mail order department is situated in Sidcup and we have 3 retail outlets at Sidcup, Lion House (Tottenham Court Rd) and Selfridges (Oxford Street). We have eight years experience of Atari products, longer than any other UK company, and are well established as the UK's No.1 Atari specialist. With a group turnover of over £9 million and in excess of 80 staff, we offer you unbeatable service and support. We provide several facilities which you will find invaluable during your Atari computing life and most of these facilities are available **ONLY FROM SILICA**. We suggest that you read through what we have to offer, before you decide where to purchase your Atari ST.

### FREE STARTER KIT - *Only From Silica*

When you purchase any Atari ST keyboard, you will not only receive the best value for money computer on the market, but you will also receive the following from Atari Corporation as part of the package:

** BASIC Language Disk * BASIC Manual * ST Owners Manual * TOS/GEM on ROM*

If you buy your ST from Silica Shop, you will also receive:

** NEOchrome Sampler - colour graphics program * 1st Word - Word Processor*

In addition, we at Silica would like to see you get off to a flying start with your new computer, so we have put together a special **ST STARTER KIT** worth over £100, which we are giving away **FREE OF CHARGE** with every ST computer purchased at our normal retail prices. This kit is available **ONLY FROM SILICA** and is aimed at providing users with a valuable introduction to the world of computing. We are continually upgrading the ST Starter Kit, which contains public domain and other licensed software, as well as books, magazines and accessories all relevant to ST computing. Return the coupon below for full details.

### DEDICATED SERVICING - *Only From Silica*

At Silica Shop, we have a dedicated service department of seven full time Atari trained technical staff. This team is totally dedicated to servicing Atari computer products. Their accumulated knowledge, skill and experience makes them second to none in their field. You can be sure that any work carried out by them is of the highest standard. A standard of servicing which we believe you will find **ONLY FROM SILICA**. In addition to providing full servicing facilities for Atari ST computers (both in and out of warranty), our team is also able to offer memory and modulator upgrades to ST computers.

**1Mb RAM UPGRADE:** Our upgrade on the standard Atari 520ST-M or 520ST-FM keyboard will increase the memory from 512K to a massive 1024K. It has a full 1 year warranty and is available from Silica at an additional retail price of only £86.96 (+VAT = £100).

**TV MODULATOR UPGRADE:** Silica can upgrade the 1040ST-F to include a TV modulator so that you can then use it with your TV set. This is an internal upgrade and does not involve any untidy external boxes. A cable to connect your ST to any domestic TV is included in the price of the upgrade which is only £49 (inc VAT). The upgrade is also available for early 520ST computers at the same price.

### THE FULL STOCK RANGE - *Only From Silica*

We aim to keep stocks of all Atari related products and our warehouse carries a stock of over £½ million. We import many software titles direct from the USA and you will find that we have new releases in advance of many of our competitors. Unlike dealers who may only stock selected titles, we have the full range. In addition, we carry a complete line of all books which have been written about the Atari ST. A range as wide as ours is something you will find is available **ONLY FROM SILICA.**

### AFTER SALES SUPPORT - *Only From Silica*

Rest assured that when you buy your ST from Silica Shop, you will be fully supported. Our free mailings give news of releases and developments. This will help to keep you up to date with new software releases as well as what's happening in the Atari market. And in addition, our sales staff are at the end of a telephone line to service all of your Atari requirements. If you purchase an ST from Silica and would like any technical advice, we have a full time technical support team to help you get the best from your computer. Because we have both the staff and the systems specifically dedicated to providing after sales service on Atari ST computers, we are confident that our users enjoy an exceptionally high level of support. This can be received **ONLY FROM SILICA**.

### FREE CATALOGUES - *Only From Silica*

At Silica Shop, we recognise that serious users require an in-depth information service, which is why we mail free newsletters and price lists to our ST owners. These are up to 48 pages long and are crammed with technical details as well as special offers and product descriptions. If you have already purchased an ST and would like to have your name added to our mailing list, please complete the coupon & return it to us. This information service is available **ONLY FROM SILICA**.

### FREE OVERNIGHT DELIVERY - *From Silica*

Most orders are processed through our computer within 24 hours of receiving them. Most hardware orders are sent by the overnight GROUP 4 courier service FREE OF CHARGE to customers within the UK. This method helps to ensure minimum delay and maximum protection.

### PRICE MATCH - *Only From Silica*

We hope that the combination of our low prices, FREE UK delivery service, FREE Starter Kit and FREE after sales support, will be enough to make you buy your Atari equipment from Silica Shop. If however, there is something you wish to purchase, and you find one of our competitors offering it at a lower price, then please contact our sales department, providing us with our competitor's name, address and telephone number. If our competitor has the goods in stock, we will normally match the offer (on a 'same product - same price' basis) and still provide you with our normal free delivery. We realise that we are not the only company who will match a competitor's price. However, if you come to us for a price match, you will also be entitled to our after sales service, including free newsletters and technical support. This makes our price match promise rather special, something you will receive **ONLY FROM SILICA**. We don't want you to go anywhere else for your Atari products. So shop at Silica, the UK's No1 Atari Specialist.

**SIDCUP** (& Mail Order) **01-309 1111**
*1-4 The Mews, Hatherley Road, Sidcup, Kent, DA14 4DX*

**LONDON** **01-580 4839**
*Lion House (1st floor), 227 Tottenham Court Rd, London, W1P OHX*

**LONDON** **01-629 1234 ext 3914**
*Selfridges (1st floor), Oxford Street, London, W1A 1AB*

## SO MUCH FOR SO LITTLE!

There is nothing that can compare with the incredible value for money offered by Atari's 520ST-FM. For only £260 (+VAT=£299), you can purchase a powerful 512K RAM computer, with a 95 key keyboard (including numeric keypad), MIDI interface, GEM, a palette of 512 colours, mouse controller, and a 512K built-in disk drive. The 520ST-FM has a TV modulator built-in, and comes with a lead to allow you to plug it straight into any domestic colour television set. The mains transformer is also built-in to the keyboard, so there are no messy external boxes. You couldn't wish for a more compact, powerful and stylish unit. Atari ST computers are now firmly established in the UK, there are nearly 500 software titles already available for a wide variety of applications and the list is growing all the time. And that's not all. When you buy your new 520ST-FM (or any Atari ST computer) from Silica Shop, you will get a lot more, including a FREE Silica ST Starter Kit worth over £100. Read the ONLY FROM SILICA section on the left, to see why you should buy your new high power, low price 520ST-FM from Silica Shop, the UK's No1 Atari Specialists. For further details of the range of Atari ST computers and the FREE Silica ST Starter Kit, complete and return the reply coupon below.

**ATARI 520ST-FM NOW ONLY £260 (+VAT=£299)**

**520ST-FM with 512K RAM & mono monitor £399** (inc VAT) **Upgrade from 512K RAM to 1024K RAM £100** (inc VAT)

## ATARI 1040ST-F - NEW PRICE

We are pleased to announce a new reduced price point on the 1040ST-F which is now available for only £499 (inc VAT). The 1040 is a powerful computer with 1Mb of RAM and also includes a built-in 1Mb double sided 3½" disk drive. The 1040 has been designed for use on business and professional applications most of which require a high resolution monochrome or colour monitor. It does not therefore have an RF modulator for use with a domestic TV set. Modulators can be fitted for £49 (inc VAT).

| | |
|---|---|
| *1040ST-F Keyboard Without Monitor* | ***£499** (inc VAT)* |
| *1040ST-F Keyboard + High Res SM125 Mono Monitor* | ***£599** (inc VAT)* |

If you would like further details of the 1040ST-F, return the coupon below.

## MEGA ST's NOW IN STOCK

For the user who requires even more RAM than the 520 or 1040 ST's offer, the new MEGA ST computers are now available. There are two MEGA ST's, one with 2Mb of RAM and the other with a massive 4Mb. Both new computers are fully compatible with existing ST's and run currently available ST software. The MEGA ST's are styled as an expandable Central Processing Unit with open architecture and a detachable keyboard. They are supplied with GEM, a free mouse controller and all extras as with the 520 or 1040. Prices are as follows:

| | |
|---|---|
| *MEGA ST 2Mb Keyboard + CPU* | ***£899** (inc VAT)* |
| *MEGA ST 2Mb Keyboard + CPU + SM125 Mono Monitor* | ***£999** (inc VAT)* |
| *MEGA ST 4Mb Keyboard + CPU* | ***£1199** (inc VAT)* |
| *MEGA ST 4Mb Keyboard + CPU + SM125 Mono Monitor* | ***£1299** (inc VAT)* |

If you would like further details of the MEGA ST's, return the coupon below.

**ATARI ST**

To: Silica Shop Ltd, Dept ATCOC 1087, 1-4 The Mews, Hatherley Road, Sidcup, Kent, DA14 4DX

## PLEASE SEND ME FREE LITERATURE ON THE ATARI ST

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

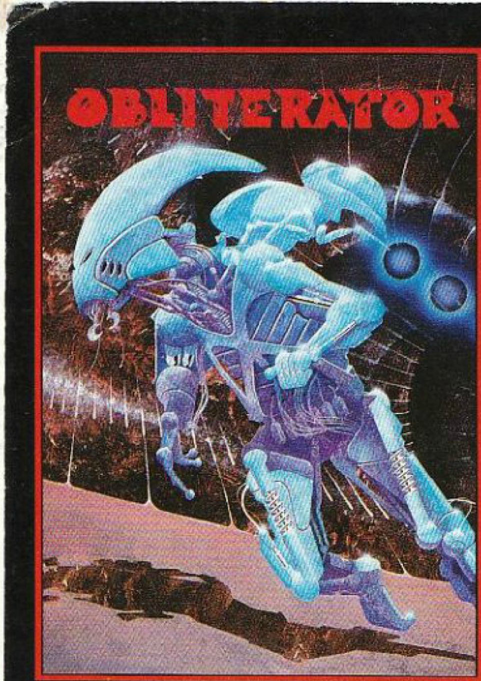
**Address:** ..........................................................................................................

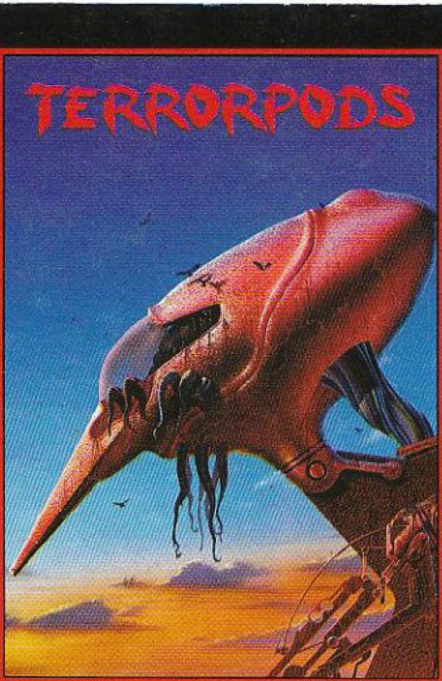
..........................................................................................................

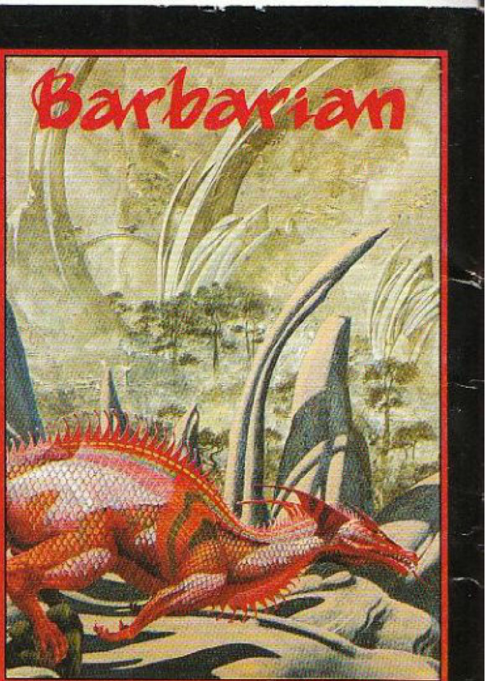
..........................................................................................................

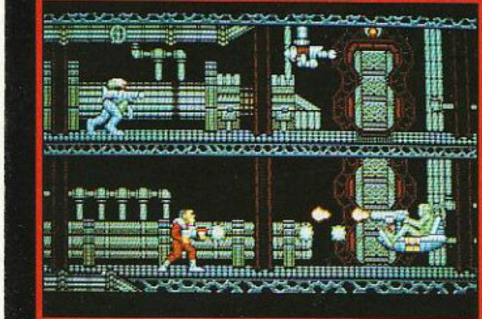
.................................................................. **Postcode:** ..............................

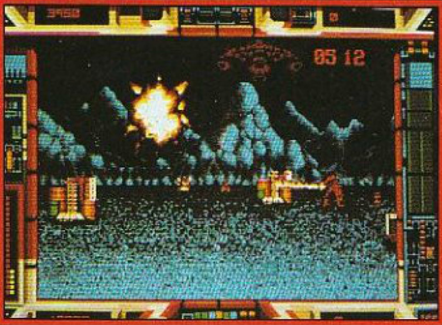
**Do you already own a computer**
**If so, which one do you own?** ..................................................................

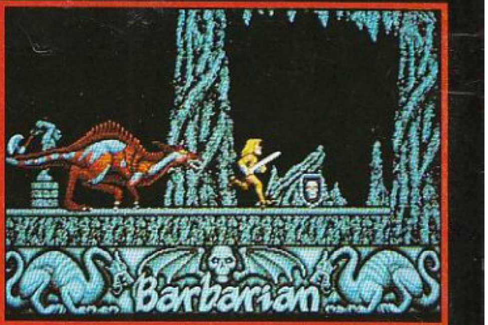












### OBLITERATOR

You are Drak the Last of the Obliterators, a genetically enhanced and awesome fighting machine, bio-engineered for the execution of incredible tasks, the ultimate solution to the most cataclysmic of predicaments.

In the voids of Federation space an alien cruiser has materialised. It is a ship of sinister and foreboding power. Federation defences have been smashed. Earth lies exposed, there is only one hope and you are it!

Summoned by the Federation council, your assignment is to use a prototype matter transporter to infiltrate the alien cruiser. Armed with a particle beam disrupter, lasers and whatever the alien habitat can provide, you must cripple the alien vessel thus enabling its destruction.

What awaits? What technological perils and strange diabolic adversaries will try to thwart you in your mission?

Can you become Drak the Last of the Obliterators? Can you survive? Can you overcome such unthinkable odds?

*COMING SOON*

### TERRORPODS

Its been a long watch . . . As the sun disappears over the horizon, the uninviting, grey bleakness of Colian becomes apparent. Following the intense heat of the day, the onset of night adds the bitterness of sub-zero temperatures to an already hostile environment, and the stark interior of the D.S.V. appears almost homelike.

Deep melancholy is suddenly smashed by the shrill scream of a siren. The status panel has gone crazy, an extraordinary array of lights flash uncontrollably. Good grief . . . what's happening?

Frantically, you turn to look at the command scanner, hunting through the mass of information before you, in a desperate attempt to decipher what has happened.

Your whole being freezes . . . It can't be! The Terrorpods . . .

*AVAILABLE NOW*

### BARBARIAN

Can you become Hegor the famous dragon-slaying, monster-mangling Barbarian?

Are you the warrior who can enter the fearful realms of the underground world of Durgan, a world terrorized by the evil Necron?

Can you handle the adventure, the frenzied attacks, the hidden traps, the gruesome death dealing monsters? . . .

Your quest; to destroy the lair of the accursed Necron. Your prize; the kingdom's crown.

Your task is awesome! You must live on your wits, conquer your innermost fears, use every skill and weapon available to you.

Hideous perils await. Can you survive? . . .

Are you Hegor the famous dragon-slaying, monster-mangling Barbarian?

*AVAILABLE NOW*

**Psygnosis**
**FREEPOST**
**Liverpool L3 3AB**
**United Kingdom**
**Tel. No: 051 236 8818**
**Fax: 051 207 4498**
**Telex: 629474**

ACCESS VISA

| | |
|---|---|
| OBLITERATOR | — £24.95 |
| TERRORPODS | — £24.95 |
| BARBARIAN | — £24.95 |
| DEEP SPACE | — £24.95 |
| ARENA | — £24.95 |
| BRATACCAS | — £24.95 |
| ARENA/BRATACCAS | — £29.95 |

*All available for 512k colour, Atari ST, Commodore Amiga.*

PSYGNOSIS