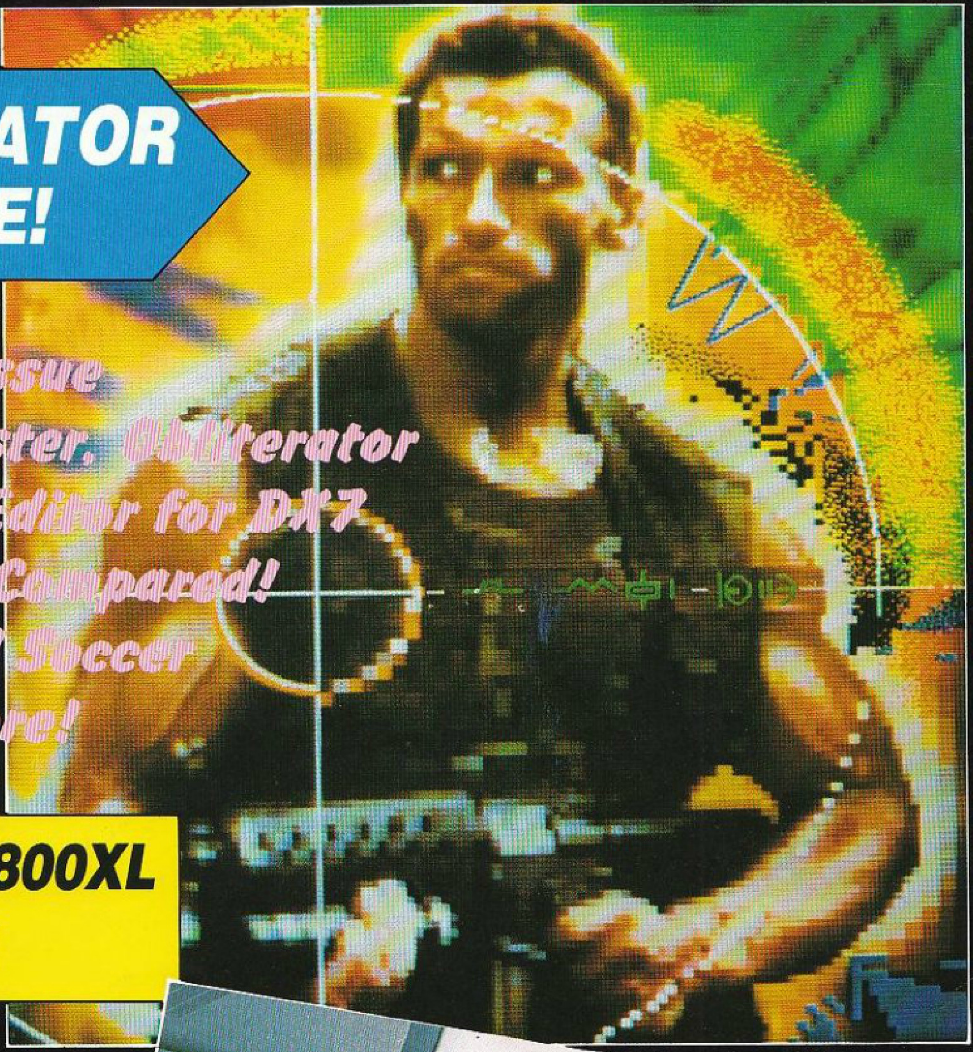
ST and XL/XE

Issue 19 Price £1.00

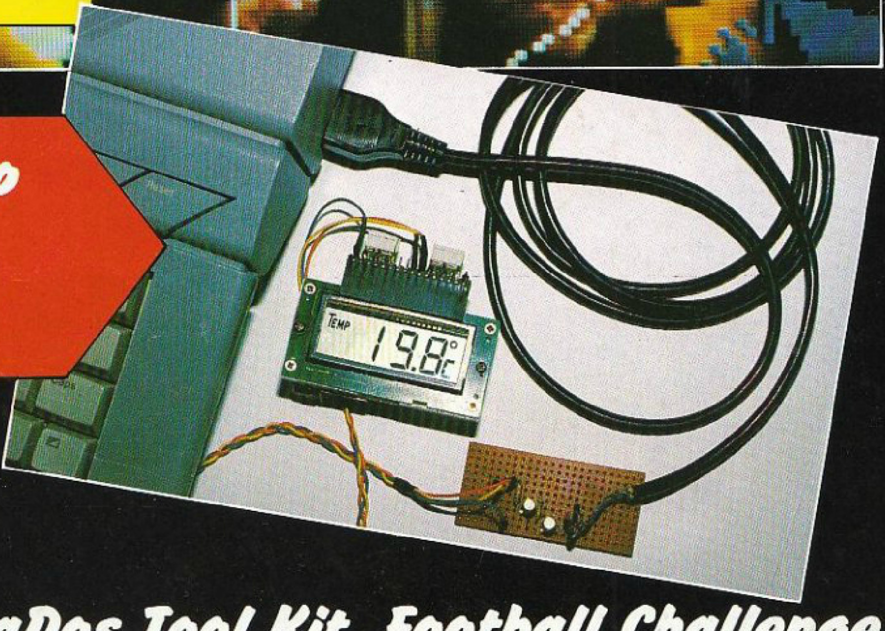
Atari Monitor



THE PREDATOR IS HERE!

Also in this Issue
Dungeon Master, Obliterator
The Russ AI Editor for DX7
Two Pascals Compared!
International Soccer
and much more!

Expand your 800XL to 256K

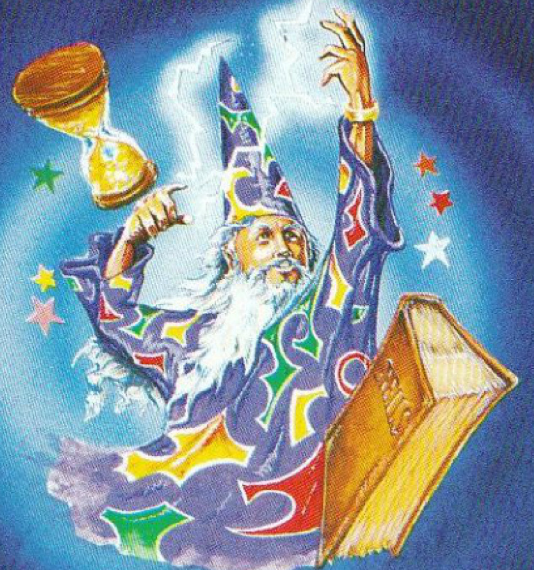


Turn your XL into a Thermometer for under £10

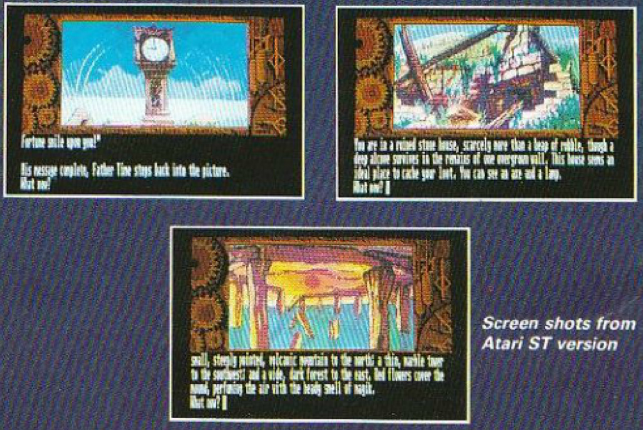
8-Bit Reviews
Expander, SpartaDos Tool Kit, Football Challenge

Time and magik

The trilogy



Interactive fiction from Level 9



Screen shots from Atari ST version

Three of Level 9's most acclaimed adventures – Lords of Time, Red Moon and The Price of Magik – come together in one package. Each has been enhanced and enlarged with more text than ever before – 60,000 mind-expanding words creating magik and mystery round every corner. There's a powerful new parser, and most disc versions include stunning digitised pictures that help to dramatically heighten the atmosphere.

What the press have said:

LORDS OF TIME: ***"Destined to become a classic"*** – *Computing with the Amstrad.*

RED MOON: ***Best Graphical Adventure or Best Adventure of the Year*** – *Zzap 64, Crash, Amtix, C&VG, CCI, and Amstrad Computer User.*

THE PRICE OF MAGIK: ***"Another superb adventure . . . their best yet"*** – *Commodore User. Also a Crash Smash.*

Spectrum (cassette) Plus 3 (disc) Commodore 64 (cassette or disc) Amstrad CPC (cassette or disc) Atari XL/XE (cassette or disc) Amstrad PCW (disc) Apple II (disc)	Atari ST Commodore Amiga Macintosh Amstrad PC, IBM PC and compatibles
£14.95	**£19.95**

MANDARIN SOFTWARE

Europa House, Adlington Park,
Adlington, Macclesfield SK10 4NP.

ENQUIRIES: 0625 878888 ORDER HOTLINE: 0625 879920

Available from all good stockists or order direct by sending a cheque or postal order made payable to Mandarin, together with your name and address. Price includes P&P. Access/Visa owners: Phone our hotline or send your card number with your order.

THE U.K. ATARI COMPUTER OWNERS CLUB
INDEPENDENT USER GROUP

CONTENTS

Write for Monitor

If you think you can write a good review, can explain clearly and precisely about a programming technique, have written a useful program others may find handy, have interesting hints and tips on commercial products you'd like to pass on, then we would love to hear from you. Write to the Editor now!

Monitor Bookshop

Introducing a new service for readers, the Monitor Bookshop. Selected books will placed in the bookshop for you to purchase from us, the books will range from beginners to expert level and should be of great benefit to you in your hobby. New books will be added every quarter and hopefully the service to you will get better and better. We've started with a small number of books (which we have vetted) from which you can make your selection.

Sad Note

We wish to express our deepest sorrow at the tragic death of Ian Martin in a road accident. Ian M was a well known figure on the 8 bit computing scene, prolific as a programmer on 6502 and Z80 machines and an Atari enthusiast and valued club member. His latest work, the game ACE 2 should appear posthumously on the ST by the time you read this. Once again, our condolences to his family and friends, he will be missed greatly.

ASSOCIATION OF ATARI USER GROUPS MEMBER

Cover: Predator from Activision.

Club Address: P.O. Box 3, Rayleigh, Essex SS6 8LR.

CREDITS

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

ADVERTISEMENTS

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

CRACKING THE CODE

Part Fifteen by Keith Mayhew

In the last part of this series we looked at how the CIO allows an independent and simple method for transferring data to or from devices. This is the level at which most programs would work for handling keyboard input, screen output, disk files, etc. This time we look at how the CIO actually implements the commands which are given to it and look at how to implement a new device handler and install it.

How CIO Executes Commands

Figure 1 illustrates the structure of the parts of the O.S. which are involved in input/output operations and also labels the main data structures used to communicate between routines.

The main interface to the O.S. from the user program is the CIO which communicates with one of the eight IOCBs at any one time. In performing the specified operation the CIO copies the relevant IOCB into an internal structure in page zero called ZIOCB at 20 hex. It then determines where the handler for the specified device resides and uses it to perform the actual input or output for that device. CIO generally modifies the ZIOCB as the operation progresses and copies the contents of the ZIOCB back into the original IOCB upon completion of the command.

As not all device handlers are resident in the system, most notably the disk drive and the RS232 interface, the O.S. employs a table to keep track of the installed handlers. This method makes it simple not only to add new device handlers at any time but also to modify or replace the operations of existing handlers.

The device handler table, HATABS at 31A hex, has room for a maximum of twelve entries, each consisting of 3 bytes. Initially the system fills in the table with its resident handlers and sets spare entries to zero. Each entry consists of the letter of the device in upper case ASCII followed by a two byte address, stored in usual reverse order, which points to the handler's vector table.

When CIO is requested to open an IOCB it searches the handler table for an entry with a matching device name. This search is performed from the end of the table so that if there is a multiple entry for a device, the one nearer to the end of the table is chosen. CIO places this index into HATABS into the user's IOCB at ICHID. If the user specified a device number, such as D2:, then CIO places this number into ICDNO. If no number is specified then CIO assumes one, thus D: is the same as D1: to the CIO. Having recorded this information from an open command means that further calls are then much faster. When an IOCB is closed its ICHID value is set to FF to indicate it is free.

Device Handlers

Each device handler, as mentioned above, has a vector table. This table consists of a list of the entry points to the handler's code as illustrated in Figure 2. What you might find rather confusing is that each address of a routine in this table is one less than the actual address. This is done because the 6502 does not have a JMP indexed-indirect instruction. To make the operation as fast as possible the CIO uses a little 'trick', it pushes the two bytes onto the stack and executes an RTS instruction. RTS expects to find the address of the jump location minus one on the stack as this is what its counterpart, JSR, puts onto the stack as a return address.

Each handler has access to the zero page IOCB, ZIOCB, its entries have the same meaning as the originating IOCB but they do not necessarily hold exactly the same values. The locations and their labels are as listed below:

20 = ICHIDZ = Handler index.
21 = ICDNOZ = Device number.
22 = ICCOMZ = Command.
23 = ICSTAZ = Status.
24 = ICBALZ = Buffer address low.
25 = ICBAHZ = Buffer address high.
26 = ICPTLZ = Put-byte routine low.
27 = ICPTHZ = Put-byte routine high.
28 = ICBLLZ = Buffer length low.
29 = ICBLHZ = Buffer length high.
2A = ICAX1Z = Auxiliary byte 1.
2B = ICAX2Z = Auxiliary byte 2.

Note that the extra 4 auxiliary bytes in the original IOCB are not copied into or out of the ZIOCB as these locations are used for other purposes internally to CIO. If a handler needs access to these values it must go to the original IOCB.

When a handler routine is called by the CIO, the X register contains the IOCB index the user originally supplied so that it is possible to access the originating IOCB. The Y register is automatically set to 92 hex which is the error code for function not implemented, thus if a routine is not used by a handler it need only perform an RTS. If the handler does provide a routine for any functions then it must set the Y register to 1 for success or a suitable error code before returning. Following are the descriptions of the functions a handler can provide.

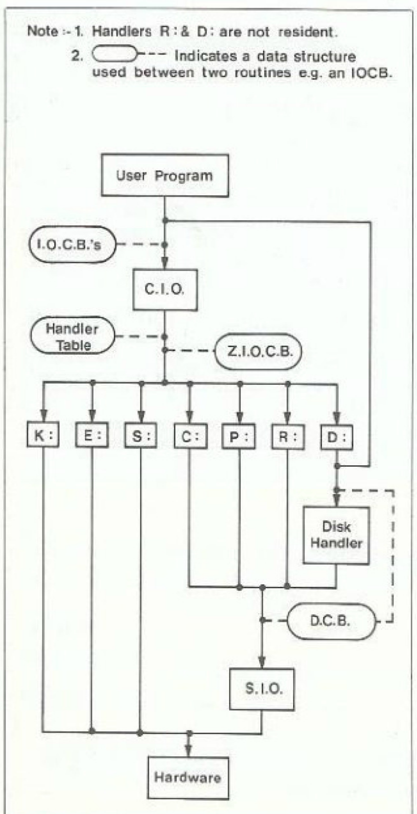


Figure 1.

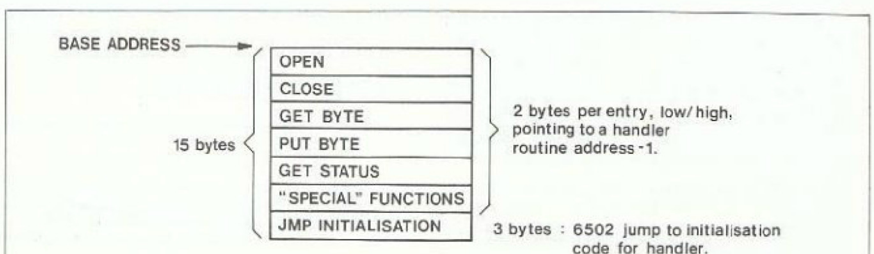


Figure 2.

Handler OPEN

The handler has to prepare for IO with its device. ICBALZ and ICBAHZ point to the filename, which will be ignored unless the device supports named files, e.g. a disk. The device number is always available to the handler from ICDNOZ if it supports more than one device. The auxiliary bytes ICAX1Z and ICAX2Z should be checked to make sure that they conform with the device. For example, the handler should not allow writing to a read-only device.

Handler CLOSE

The handler should make sure that all pending operations are completed, such as writing data to a device, before returning whereupon CIO will mark the IOCB as free.

Handler GET BYTE

The handler should return the next byte from the device in the A register.

Handler PUT BYTE

The handler should put the byte in the A register out to the device.

Handler GET STATUS

The handler should respond by filling any necessary bytes into the four bytes starting at DVSTAT at 2EA hex. For most handlers this function will not be of any use as normal status of operations are returned via the Y register and subsequently stored in the IOCB by the CIO. If this is the case then the handler should always return one in the Y register indicating this routine succeeded. It will not have to alter the status bytes as they are device dependent.

Handler SPECIAL

If the CIO finds a command byte with a value of 0D hex or greater then it calls the SPECIAL entry of the handler with the contents of the ZIOCB the same as the originating IOCB. If the handler does not support any extended commands then it should just return. An example of a special command is the DRAWTO command of the display handler; the handler can determine which special command is requested by looking at ICCOMZ.

For SPECIAL commands or a GET STATUS command the CIO supports an implied open mode. This allows such a command to be performed on a closed IOCB. CIO first opens the IOCB, performs the action needed and then closes it before returning. The buffer address must point to a suitable device/filename and the auxiliary bytes must be set as for a normal open command if the implied open method is to succeed.

Handler INITIALISATION

The last entry in a handler's vector table is a JMP instruction to code which is called for initialisation at reset and power-up. Unfortunately, the person who wrote CIO forgot to call these vectors at all! The initialisation entry should just return one in the Y register. The problem caused by this omission is that after a reset the handler table which keeps the device names is re-built, so any devices which were added are now removed and hence their names are unknown to the system. You can still get round this by patching into the vector for a reset and then re-installing the device name.

The BASIC Complication

The people who wrote ATARI BASIC managed to get a loophole put into the CIO to enable faster writing operations from BASIC, particularly for byte-at-time output to a device. This loophole is the put-byte vector in each IOCB: ICPTL & ICPTH, and their counterparts in the ZIOCB: ICPTLZ & ICPTHZ. The put-byte vector of the IOCB is a duplicate of the entry in the handler vector table. When an IOCB is opened, CIO copies the vector from the handler to the IOCB; when closed the entry is set to point to a routine which returns an error code for IOCB-not-open.

The presence of this put-byte vector is usually of little significance but can cause unexpected results if you are not careful. The main thing is that you should not rely upon the values in the ZIOCB for a put-byte operation and look at the originating IOCB instead. The handler should also check that a write operation is not being performed to the device if it was opened in read-only mode.

As the operating system does not use this put-byte vector you only have to consider the problems if the device will be used from within the BASIC environment.

Overview of Serial Input Output

As can be seen from Figure 1, some handlers interface directly with the hardware whereas others use the sub-system known as the Serial Input Output (SIO). It is the SIO which handles all transfers of data to external devices connected via the 13-pin serial port, such as disk drives, printer or cassette.

Communication between a handler and SIO is done via the Device Control Block (DCB) data structure. Usually you would not set up values in the DCB and call SIO directly but use the appropriate handler via a call to CIO and let the handler deal with the complications for that particular device.

The exception to this structure is the disk drive which has a resident disk handler which provides a very low level access to a disk on a sector basis. The user has direct access to this handler but its main purpose is to read and write sectors for the CIO non-resident device handler D:. This is what is referred to as the Disk Operating System (DOS) and provides a multiple-file structure for a disk. To maintain these files DOS allocates some sectors of the disk for a directory of the names of the files and where they are actually stored on the disk.

A New Device Handler

Putting theory into practice, we will now look at how a new device handler can be written. Listing 1 is the assembly code for a device handler called B: which provides a buffer in memory for data to be stored in and read out of. With B: you can copy a small file into it from DOS and then copy it onto, say, another disk. It is not a RAM disk in the sense that you cannot write more than one file to the device. However, a special feature of B: is that it is in fact a queue.

A queue allows you to write data to it and then, while the device is still open, start reading the data already written. If the data being written reaches the end of the buffer it can wrap around to the beginning if some data has already been read. In effect this means that an infinite amount of data can be written to the device as long as data is also read. This could be useful in a program which generates a lot of intermediate data and wants somewhere to write it before another part of the program reads the data again.

Listing 2 is a BASIC program which reads the device handler into page 6, it is initialised with X=USR(1536). The device uses 1K of memory starting at 6000 hex as the circular buffer.

The program starts by finding a spare entry in the handler table, HATABS, where it can insert the character 'B' and the address of the vector table VECTAB: this installs the device in the CIO. There are four pointers which are initialised for the device, these are: BUFSTRT which points to the start of the buffer; BUFLAST which points to the end of the buffer; BUFPUT which points to where the next character will be put; BUFGET points to where the next character will be read from. The number of bytes in the buffer is held in BUFLEN, this is initially zero.

The handler only has code for OPEN, GET & PUT. The rest of the operations are not needed and so are set to suitable return points in the program. The OPEN function performs no function if the auxiliary byte does not specify a write operation: this is because if there is data in the buffer, the pointers do not need to be changed. For an open with a write, the pointers are reset and any previous data is lost.

The GET function returns the next available byte assuming that BUFLEN is non-zero, i.e. there are bytes to be read. It decrements BUFLEN if a byte was read and moves BUFGET to the next character. If BUFLEN was zero then

```
0100 ; Handler for a buffer device B:
0110 ICAX1Z  =      $2A      ;Zero page auxiliary 1.
0120 HATABS  =      $031A    ;Handler table.
0130 ICAX1   =      $034A    ;IOCB auxiliary 1.
0140 BUFPGS  =      4        ;Number of pages for buffer.
0150 WRITE   =      8        ;Open for write bit.
0160 RDONLY  =      135      ;Error - read only.
0170 ENDFILE =      136      ;Error - end of file.
0180 DSKFULL =      162      ;Error - disk full.
0190         *=     $CB
0200 BUFGET  *=     *+2      ;Buffer get pointer.
0210 BUFPUT  *=     *+2      ;Buffer put pointer.
0220 BUFLEN  *=     *+2      ;Number of bytes in buffer.
0230 BUFADDR =      $6000    ;Address of buffer.
0240         *=     $0600
0250         PLA
0260         LDX    #33      ;Search HATABS from end.
0270 NEXTDEV LDA    HATABS,X
0280         BEQ    EMPTY    ;Spare slot?
0290         DEX
0300         DEX
0310         DEX
0320         BPL    NEXTDEV ;Next device entry.
0330         RTS             ;No room found!
0340 EMPTY   LDA    #'B'     ;Name of handler.
0350         STA    HATABS,X ;Point to vector table.
0360         LDA    #VECTAB&$FF
0370         STA    HATABS+1,X
0380         LDA    #VECTAB/256
0390         STA    HATABS+2,X
0400         LDA    #$00     ;Buffer address low.
0410         STA    BUFSTRT ;Start of buffer.
0420         STA    BUFGET   ;Get pointer.
0430         STA    BUFPUT   ;Put pointer.
0440         LDA    #$FF
0450         STA    BUFLAST ;End of buffer low.
0460         LDA    #0       ;Set length to zero.
0470         STA    BUFLEN
0480         STA    BUFLEN+1
0490         LDA    #BUFADDR/256 ;Buffer address high.
0500         STA    BUFSTRT+1
0510         STA    BUFGET+1
0520         STA    BUFPUT+1
0530         CLC
0540         ADC    #BUFPGS-1 ;Add number of pages.
0550         STA    BUFLAST+1 ;End address high.
0560         RTS             ;Installed.
0570 ;Handler OPEN.
0580 OPEN    LDA    ICAX1Z   ;Get auxiliary byte.
0590         AND    #WRITE   ;See if write mode.
0600         BEQ    OPENOK   ;No.
0610         LDA    BUFSTRT
0620         STA    BUFGET   ;Reset get & put.
0630         STA    BUFPUT
0640         STA    BUFLEN   ;Reset length.
0650         STA    BUFLEN+1
0660         LDA    BUFSTRT+1 ;And high bytes.
0670         STA    BUFGET+1
0680         STA    BUFPUT+1
0690 OPENOK  LDY    #1
0700         RTS
0710 ;Handler GET.
0720 GET     LDA    BUFLEN+1 ;Any bytes in buffer?
0730         BNE    GETOK
0740         LDA    BUFLEN
0750         BEQ    EOF      ;No - end of file.
0760 GETOK   LDY    #0       ;Get byte.
0770         LDA    (BUFGET),Y
0780         PHA             ;Save it.
0790         LDX    #BUFGET ;Increment GET pointer.
0800         JSR    INCRMNT
0810         JSR    BUFWRAP ;Wrap atound if necessary.
0820         LDX    #BUFLEN ;Decrement byte count.
0830         JSR    DECRMNT
0840         PLA             ;Restore byte.
0850         LDY    #1       ;Good status.
0860         RTS
0870 EOF     LDY    #ENDFILE ;End of file status.
0880         RTS
0890 ;Handler PUT.
0900 PUT     TAY             ;Save byte.
0910         LDA    ICAX1,X ;Make sure write allowed!
0920         AND    #WRITE
0930         BEQ    NOWRITE ;Read only.
0940         LDA    BUFLEN+1 ;Room in buffer?
0950         CMP    #BUFPGS
0960         BNE    PUTOK
0970         LDA    BUFLEN
0980         BEQ    FULL     ;No - buffer full.
0990 PUTOK   TYA             ;Restore byte.
1000         LDY    #0       ;Save it.
1010         STA    (BUFPUT),Y
1020         LDX    #BUFPUT ;Increment put pointer.
1030         JSR    INCRMNT
1040         JSR    BUFWRAP ;Wrap around if necessary.
1050         LDX    #BUFLEN ;Increment byte count.
1060         JSR    INCRMNT
1070         LDY    #1       ;Good status.
1080         RTS
1090 NOWRITE LDY    #RDONLY ;Read only status.
1100         RTS
1110 FULL    LDY    #DSKFULL ;"Disk" full status.
1120         RTS
1130 ;Increment pointer at X.
1140 INCRMNT INC    0,X      ;Low byte.
1150         BNE    INCRET
1160         INC    1,X      ;High byte.
1170 INCRET  RTS
1180 ;Decrement pointer at X.
1190 DECRMNT DEC    0,X      ;Low byte.
1200         LDA    0,X
1210         CMP    #$FF
1220         BNE    DECRET
1230         DEC    1,X      ;High byte.
1240 DECRET  RTS
1250 ;Wrap pointer to start of buffer.
1260 BUFWRAP LDA    1,X      ;High byte.
1270         CMP    BUFLAST+1
1280         BCC    NOWRAP   ;Less than.
1290         BNE    WRAP     ;Greater than.
1300         LDA    0,X      ;Low byte.
1310         CMP    BUFLAST
1320         BCC    NOWRAP   ;Less than.
1330         BEQ    NOWRAP   ;Equal to.
1340 WRAP    LDA    BUFSTRT ;Wrap to start of buffer.
1350         STA    0,X
1360         LDA    BUFSTRT+1
1370         STA    1,X
1380 NOWRAP  RTS
1390 ;Vector table for device.
1400 VECTAB  .WORD  OPEN-1   ;Open.
1410         .WORD  RETOK-1  ;Close.
1420         .WORD  GET-1    ;Get.
1430         .WORD  PUT-1    ;Put.
1440         .WORD  RETOK-1  ;Status.
1450         .WORD  RETURN-1 ;Special.
1460         JMP    RETOK    ;Initialisation.
1470 RETOK   LDY    #1       ;Good status.
1480 RETURN  RTS
1490 BUFSTRT *=     *+2      ;Pointer to start of buffer.
1500 BUFLAST *=     *+2      ;Pointer to end of buffer.
```

Listing 1.

end-of-file status is returned.

The PUT function behaves similarly to GET except that it will only insert a byte if BUFLEN is not at its maximum value. If the buffer is full then the DOS error status disk-full is returned.

The get and put functions use the routines INCRMNT and DECRMNT to either add one or take one from a pointer. These routines take the address of a pointer in the X register and access the bytes using 0,X and 1,X. This makes the routines general purpose and saves duplicating code in the program for each pointer. The routine BUFWRAP also takes the address of a pointer in the X register and sets the pointer to the start of the buffer if it is currently past the end of the buffer.

Note that the PUT routine performs a test on the IOCB's auxiliary byte to ensure that BASIC is not performing a direct put call while the IOCB is only opened for reading!

There are two problems with the way this device has been written. The first is that the buffer is in a place in memory that could easily be overwritten by another program such as BASIC. However, it is difficult to provide a completely safe area of memory! The second problem is that a system-reset will remove the device from the handler table and will need to be called again to re-install itself.

Next Time

That is all for this time. In the next part we will examine the key features of each of the system handlers and also explain how the two problems mentioned above could possibly be solved.

```
QZ 10 DIM HEX$(16)
CV 20 LINE=10000:TRAP 100:J=0:START=1536
VA 30 READ HEX$,CHKSUM:SUM=0
AA 40 FOR I=1 TO 15 STEP 2
ZG 50 D1=ASC(HEX$(I,I))-48:D2=ASC(HEX$(I
   +1,I+1))-48
KT 60 NUM=((D1-7*(D1>16))*16+(D2-7*(D2>1
   6)))
LW 70 SUM=SUM+NUM:POKE START+J,NUM:J=J+1
   :NEXT I
LY 80 IF SUM=CHKSUM THEN LINE=LINE+10:GO
   TO 30
IN 90 ? "Checksum error on this line:"
VO 95 LIST LINE:END
YS 100 PRINT "Data in memory."
YQ 10000 DATA 68A221BD1A03F006,763
NZ 10010 DATA CACACA10F660A942,1199
DI 10020 DATA 9D1A03A9D99D1B03,759
LY 10030 DATA A9069D1C03A9008D,673
JI 10040 DATA EB0685CB85CDA9FF,1339
FF 10050 DATA 8DED06A90085CF85,1026
YL 10060 DATA D0A9608DEC0685CC,1193
FI 10070 DATA 85CE1869038DEE06,856
NG 10080 DATA 60A52A2908F012AD,783
RG 10090 DATA EB0685CB85CD85CF,1255
SY 10100 DATA 85D0ADEC0685CC85,1226
DO 10110 DATA CEA00160A5D0D004,1048
HW 10120 DATA A5CFF016A000B1CB,1174
ZR 10130 DATA 48A2CB20AA0620BC,865
BT 10140 DATA 06A2CF20B10668A0,854
MZ 10150 DATA 0160A08860A8BD4A,920
FR 10160 DATA 032908F01FA5D0C9,897
MQ 10170 DATA 04D004A5CFF01898,1004
UI 10180 DATA A00091CDA2CD20AA,1079
XH 10190 DATA 0620BC06A2CF20AA,803
XF 10200 DATA 06A00160A08760A0,814
IY 10210 DATA A260F600D002F601,961
FF 10220 DATA 60D600B500C9FFD0,1155
GS 10230 DATA 02D60160B501CDEE,938
EJ 10240 DATA 069015D009B500CD,774
LN 10250 DATA ED06900CF00AADEB,1057
PY 10260 DATA 069500ADEC069501,720
RR 10270 DATA 604006E7065B067C,624
TD 10280 DATA 06E706E9064CE806,796
ZP 10290 DATA A00160,257
```

Listing 2.

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'.

For sale! 800XL, 1050 W/P, 1010, Touch Tablet, 150+ games L/S box, books and magazines. Phone Jon (09747) 212.

For sale! 1029 printer, £70 plus Atariwriter. 1020 plotter, £40. 1010 recorder, £15. The Music Studio, £10. Games at £7 each, Ballblazer, Koronis Rift, The Eidolon, Summer Games, Mercenary, The Second City, Boulderdash, Conan, Mr. Robot, Hitch Hikers Guide, Transpack II. Tel: 683 4087 (evenings).

Would any readers who are involved in FORTH on any Atari machine (ABAQ development sites please write) and who are interested in pooling ideas and experiences, please write to: David Thompson, 204 Hampstead Road, Benwell, Newcastle upon Tyne NE4 8PT.

Help! I need some assistance with K Data and K Word. I would like to hear from anyone with experience of these products. Write to WO2 J. Edwards, QMs Dept., 3rd Batallion, The Queen's Regiment, BFPO 808.

Atari 400/800/XL/XE original games for sale on tape, disk and cartridge. Send SAE for a list to Adrian B, 8 Healy, Lakeside, Tamworth, Staffs.

US Doubler for sale, unused, surplus to requirements, £15. Write to Cutler, 44 Water Street, Great Harwood, Lancs BB6 7QR.

Upgrade your 800XL 64K to 256K (130XE) with Yorky Expansion box, £80 ono, brand new, unused, complete with instructions. Write to Mr. S. M. Young, 94 Mangravet Ave, Maidstone, Kent ME15 9BE.

Atari ST/Mapsat weather satellite project software. Demo disk £3, requires computer only. Refundable against full program and manual £19.95 inclusive. GEM based machine code, data and processed screens (Degas) to disk. Orbit forecast, construction tips, interfacing and more. Contact Les Kaye, Fieldvale, Park Lane, Snitterfield, Warwicks CV37 0LS.

For sale! 1029 printer as new, spare ribbon and paper. Moved to ST. Offers to B. Perrie, 19 Cheviot Crescent, Dundee DD4 9QN. Delivery may be arranged.

For sale! Magazines, books and hardware. Antic, Analog, Page 6, Monitor, send SAE for list. Mr. H. Field, The Chimes, Bells Lane, Hoo, Nr. Rochester, Kent ME3 9HT.

Wanted! Hardware/software to use 800XL for RTTY/AMTOR via amateur radio. Contact Bill Hurley, 20 Chestnut Grove, Caherdavin Lawn, Limerick, Eire. Phone Limerick 52413.

Pen Pals. Atari 8 bit disk users anywhere! Swap adventure tips and other information, etc. Write to Kev, 48 Sussex Street, Bishop Auckland, Co Durham DL14 8JD.

Pen Pals required! ST owners willing to help a beginner please write to Duncan Kay, 21 Cripplegate, Southminster, Essex CM0 7DW.

Contact. I wish to contact any Atari users in the Leighton area. Nigel Edwards, 35 Meadow Way, Leighton Buzzard LU7 8XN, tel: 382207.

Magazines/books: Page 6 numbers 2 to 31, Analog 49 and 50, Monitor 7 to 17, Computer Animation Primer, Revised Mapping the Atari. All for £35. Phone George 031 332 5158.

DIGITAL THERMOMETER INTERFACE

By Ron Levy

This article describes how to interface a digital thermometer to your Atari computer, and it will work with any of the 8 bit Atari machines. The thermometer I have used is a neat little module sold by Maplin Electronics, it has an LCD digital display, and can operate in either centigrade or fahrenheit. As well as this the module itself can also work as a clock (although the time cannot be output to a computer).

The range on this module is quite good, and although the accuracy is not amazingly high, the resolution is to a tenth of a degree, so it is very good at measuring temperature changes. This, together with its fast response time makes it particularly good for use in the laboratory, where the standard mercury thermometer's sluggish response (and hard to read scale!) makes it difficult to monitor temperatures during experiments. In fact, now you can use your computer to record the temperature at precise time intervals, leaving you free to concentrate on other aspects of your task.

The Thermometer

The module has an edge connector of 0.1 inch pitch, with 16 contacts, to provide the functions available. Table 1 shows the purpose of each of these pins, and Diagram 1 shows the edge connector. Some of these are output functions (such as over or under temperature indication and alarm), and some are input lines to control how the module operates. The input lines are operated by connecting them to the positive supply (pin 16) via a suitable switch. Without any connections being made the module operates as a thermometer in centigrade, with a display update once every ten seconds.

If you look at pins 9 and 10, you will see that these two provide the signal needed to connect the thermometer to a computer. Pin 9 is the serial data line, and pin 10 is the clock line. The thermometer sends data out each time it updates its display, which is either once every ten seconds, or once every second if pin 5 is connected to the positive line (pin 16).

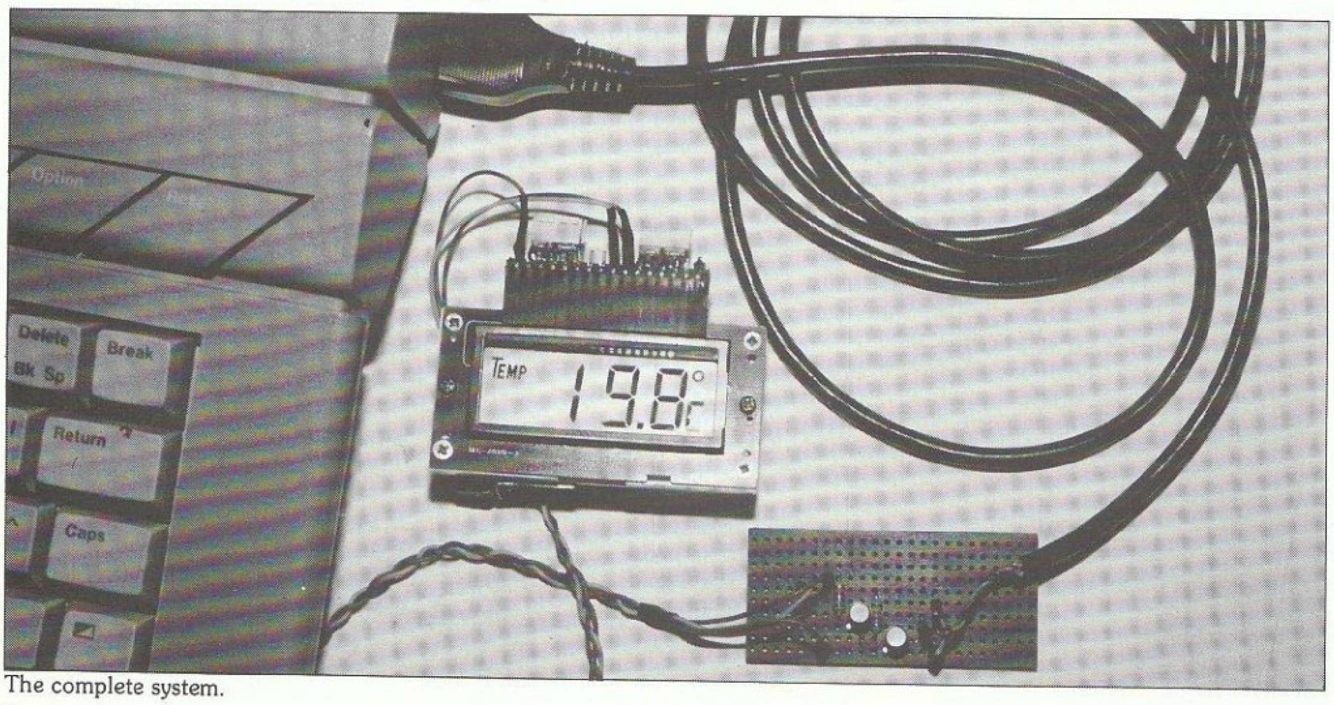
Interfacing the Module

So how does the thermometer transmit the temperature through the two lines? Well take a look at Diagram 2. What happens is that every time the temperature is updated on the module's display thirteen pulses appear at the data line (pin 10). During this time, the state of the data line (pin 9) changes, corresponding to the thirteen bits of data representing the temperature. In other words, each time the clock pin is in a 'HIGH' state, there is a valid data bit on the data line.

BCD

Binary coded decimal (BCD) might seem like an odd way to represent a four digit display, but believe it or not there is a method in the designers madness!

This data does in fact represent a three and a half digit BCD number. The first bit on its own is used to represent either the sign if the module is switched to centigrade (high means negative value,



The complete system.

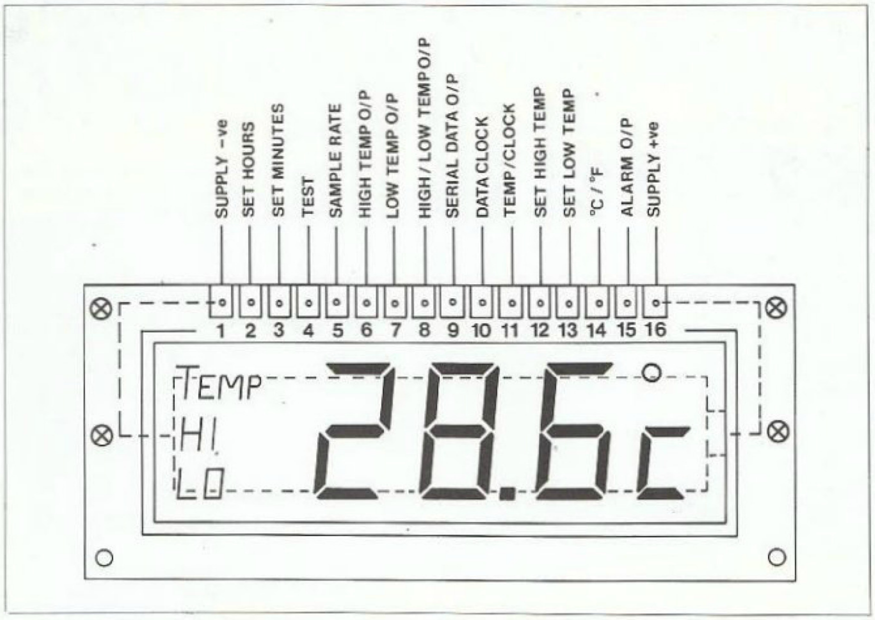


Diagram 1. Module edge connections.

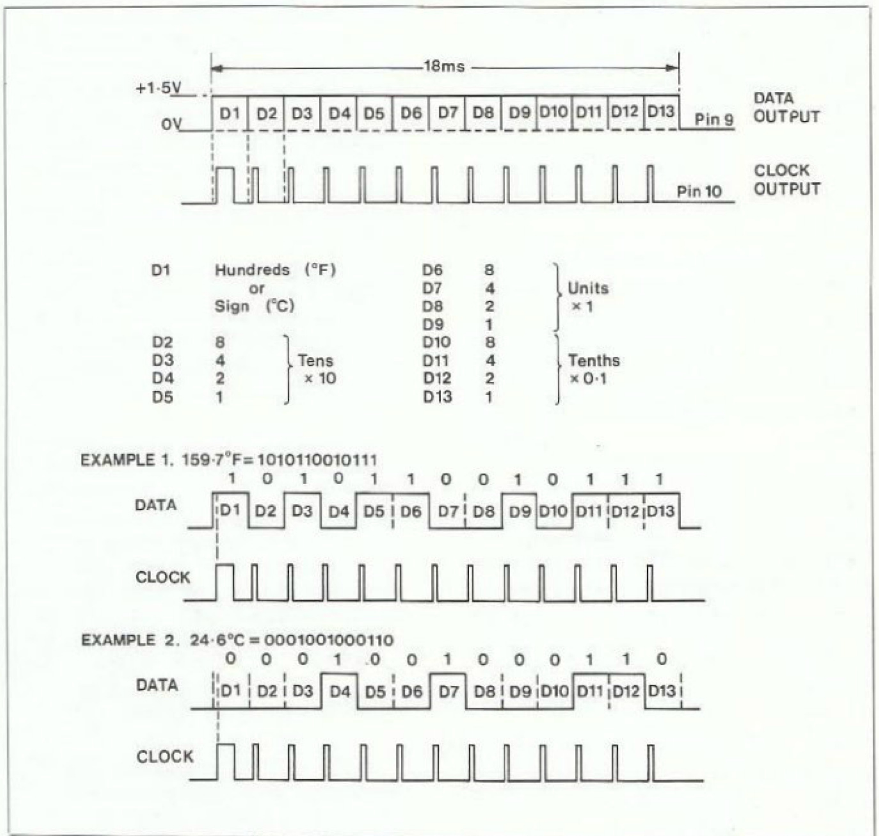


Diagram 2. Serial output data signals.

and low means positive value) or if the module is operating in fahrenheit it indicates the hundreds value. The next four bits represent the 'tens' digit, and the following four bits after that are the units. Since the decimal point always stays in the same place on the display it is not transmitted. Finally, the last four bits denote the number of tenths of a degree.

Somehow we have to read these two lines and decipher the information in the 13 data bits. The first problem with this is that we have to be sure that we actually start with the first bit (D1 on our diagram), since if a program starts sampling half way through the pulse stream we would get all kinds of strange results! If you look a little more closely at Diagram 2 you may notice that the first clock pulse is a good deal longer than the rest, this serves as our indicator, and we have to measure its length somehow in our interface program. Now take a look at Diagram 3. This shows two of the clock bits in more detail, and it should give you some idea of the time scales involved. The thirteen pulses are spaced one thousandth of a second apart, and each pulse (except the first) is just an eighth of this in length!

Edge Connections
1. Negative of 1.5V cell.
2. Set hours.
3. Set minutes.
4. Test point (activates all segments on the display when pin 2 is high).
5. Sampling rate. With switch closed display is updated once every second, with switch open display is updated once every ten seconds. Note that the set points only function in the ten second mode.
6. When the high temperature set point is reached, this output goes high for at least one minute. It remains high until the temperature falls below the set point again.
7. When the low temperature set point is reached, this output goes high for at least one minute. It remains high until the temperature climbs above the set point again.
8. When either the high or low temperature set point is reached this output pulses high for one second.
9. Serial data output.
10. Serial data clock output.
11. With switch open the temperature is displayed, with switch closed clock is displayed.
12. Set high temperature.
13. Set low temperature.
14. With switch open temperature is displayed in centigrade, switch closed gives fahrenheit.
15. When either the high or low set point is exceeded a 4kHz is output for 6 seconds.
16. Positive of 1.5V cell.

Table 1.

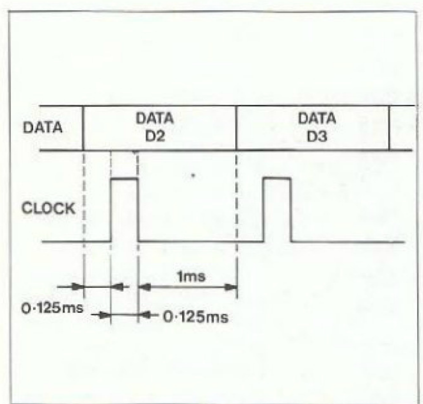


Diagram 3. Data pulses.

At this stage you may realise the second major problem - BASIC just is not fast enough to catch these pulses! Because of this we must resort to machine code to read in the data. Take a look at the machine code program in Listing 1. If you understand machine code you may find it helpful to study how this works, but if you don't, then there is no need to panic since I have converted this to data statements for a basic program.

```
0100 .OPT NOLIST
0110 PORTA   =$D300    Port A address.
0120 TRIG0   =$D010 Trigger zero reg.
0130 GRACTL  =$D01D     Trigger latch.
0140 IRQEN   =$D20E    Interrupt mask.
0150 POKMSK  =$10          OS shadow.
0160 AUDF1   =$D200    Channel 1 freq.
0170 STIMER  =$D209      Start timers.
0180 VTIMR1  =$210     Timer 1 vector.
0190 RESULT  =$D4      Result pointer.
0200 TFLAG   =$D6          Timer flag.
0210 TEMP    =$D7        Work storage.
0220 ;
0230         *=$600
0240 ; Load the result pointer.
0250         PLA
0260         PLA
0270         STA RESULT+1
0280         PLA
0290         STA RESULT
0300 ;
0310 ; Set up the timer interupt.
0320         LDA #INTVEC&255
0330         STA VTIMR1
0340         LDA #INTVEC/256
0350         STA VTIMR1+1
0360 ;
0370 ;
0380 ; Get the first data bit, and
0390 ; check for long clock pulse.
0400 BIT1    LDY #$00
0420         STY TFLAG Reset the flag
0440 ;
0450 BIT1B   LDA TRIG0  Await first
0460         BNE BIT1B   clock pulse.
0470 ;       Now enable interupt.
0480         SEI
0490         LDA POKMSK
0500         ORA #$01
0510         STA POKMSK
0520         STA IRQEN
0530 ;
0540         LDA #25  Load the timer.
0550         STA AUDF1
0560         STA STIMER  Start timer.
0570         CLI
0580 ;
0590         LDA PORTA  Read the
0600         AND #$01       data line.
0610         BEQ BIT1C
0620         LDA #$20   'Space' char.
0622         BNE BIT1D
0630 BIT1C   LDA #$2D   'minus' sign.
0640 BIT1D   STA (RESULT),Y Store it.
0650 ;
0660 WAITF   LDA TFLAG Loop until
0670         BEQ WAITF interrupt done
0680         AND #01  If not first,
0690         BNE BIT1  then re-start.
0700 ;
0710 ;
0720 ; Read the last 12 data bits.
0730 PAUSE   LDA TRIG0 Wait for clock
0740         BEQ PAUSE      to settle.
0750         INY
0760 RESETX  LDX #04
0770         STX GRACTL
0780         LDA #$00
0790         STA TEMP
0800 ;
0810 ;      Wait for clock to go high.
0820 READNXT LDA TRIG0
0830         BNE READNXT
0840         LDA #0     disable the
0850         STA GRACTL         latch.
0860 WAITHI  LDA TRIG0   Wait for
0870         BEQ WAITHI    clock off.
0880         LDA #04    Enable the
0890         STA GRACTL         latch.
0900 ;
0910         LDA PORTA Read the data.
0920         EOR #01
0930         LSR A
0940         ROL TEMP
0950         DEX
0960         BNE READNXT Get next bit
0970         LDA TEMP
0980 ;
0990         CLC   Convert to ASCII
1000         ADC #$30         numeric.
1010         STA (RESULT),Y
1020         INY
1030         CPY #04
1040         BNE RESETX
1050 ;
1060         STA (RESULT),Y
1070         DEY
1080         LDA #$2E   An ASCII dot.
1090         STA (RESULT),Y
1100         LDA #00       Reset the
1110         STA GRACTL         latch.
1120 DONE    RTS           Final exit.
1130 ;
1140 ;
1150 ; Timer interupt routine.
1160 INTVEC  SEI
1170         LDA POKMSK
1180         AND #254
1190         STA POKMSK
1200         STA IRQEN
1210 ;
1220         LDA TRIG0
1230         ORA #04
1240         STA TFLAG
1250         PLA
1260         CLI
1270         RTI
1280 ;      End of interupt routine.
```

Listing 1.

The Machine Code Program

The program monitors the clock line, and waits for the arrival of the first pulse. Once it has detected this it must determine whether this pulse is the first one by checking its length. It does this by using an interrupt routine triggered by one of the hardware timers counting down to zero. If the pulse is the first one then the data line is read and stored into the output string as a 'minus' sign or a blank. Assuming that the thermometer is switched to centigrade, this is the sign of the value. The next twelve bits are then read and interpreted by the program, which even converts each BCD part into its ASCII character.

Another minor problem I encountered was that the program occasionally missed a bit. This was because the clock pulses are so short (0.125 milliseconds) that they can be missed by the program if they occur whilst the computer is servicing one of its longer interrupts, or if ANTIC has stalled the processor to access the screen memory. To get around this the clock line is connected to one of the joystick trigger lines, and the program puts the trigger input into 'latch' mode so that it 'remembers' a pulse until it is reset.

The result of all this is a string which will hold the number of degrees centigrade. This string can be printed on the screen as it is, but if you want to do any calculations with the value, or plot a graph, you can convert the string to a numeric variable using the VAL command.

Listing 2 is the basic framework you will need to incorporate into your own program for it to read the temperature. The machine code is read from data statements at the start of your program, and it is placed into the page 6 area of memory where it is quite safe. When your program needs to read the temperature it does so using the following command:

Z = USR(1536,ADR(T$))

The routine then reads the temperature via joystick port 1, and places it (as a decimal string) into the variable T$.

As an example of using this, I have included Listing 3. This reads the temperature from the joystick port and uses it to plot a simple graph. Since the thermometer transmits the temperature at precise time intervals (either one second or ten seconds) this can be used for any timing that your program may require. You may notice that Listing 2 asks you what time interval, in seconds, you want between each plot on the screen, and to do this I have simply issued calls to the temperature routine and just ignored the results I did not want.

```
EH 0 REM ******************************
QP 1 REM * Prog. 2 - Reads Thermometer.*
EM 2 REM * -------   ----------------- *
CI 3 REM * Reads the serial data from  *
PO 4 REM * the digital thermometer     *
IX 5 REM * interface via joystick port *
YJ 6 REM * 1. Result is placed into    *
KF 7 REM * the area of memory specified*
LM 8 REM * in the USR call (ie T$).    *
EQ 9 REM ******************************
ZF 10 DIM T$(5)
UE 20 T$="*****"
VS 23 REM interface program from data.
BA 30 REM
WW 40 REM This reads the machine code
VP 50 REM interface program from data.
VP 60 FOR I=1536 TO 1536+158
NJ 70 READ X:POKE I,X
IW 80 NEXT I
QM 100 REM
ZI 101 REM This loop reads the serial
WQ 102 REM data at joystick 1.
IA 110 Z=USR(1536,ADR(T$)):T=VAL(T$)
PC 120 PRINT T
LY 190 GOTO 100
SF 199 REM
AS 32750 REM ....Themometer I/F Code....
ME 32751 DATA 104,104,133,213,104,133,212
   ,169,139,141,16,2,169,6,141,17,2,160,0
   ,132,214,173,16,208,208,251,120,165
QD 32752 DATA 16,9,1,133,16,141,14,210,16
   9,25,141,0,210,141,9,210,88,173,0,211,
   41,1,240,4,169,32,208,2,169,45,145,212
YQ 32753 DATA 165,214,240,252,41,1,208,20
   5,173,16,208,240,251,200,162,4,142,29,
   208,169,0,133,215,173,16,208,208,251
GP 32754 DATA 169,0,141,29,208,173,16,208
   ,240,251,169,4,141,29,208,173,0,211,73
   ,1,74,38,215,202,208,225,165,215,24
KU 32755 DATA 105,48,145,212,200,192,4,20
   8,204,145,212,136,169,46,145,212,169,0
   ,141,29,208,96,120,165,16,41,254,133
LZ 32756 DATA 16,141,14,210,173,16,208,9,
   4,133,214,104,88,64
```

Listing 2.

```
EH 0 REM ******************************
QI 1 REM * Prog. 3 - Temperature Graph.*
EM 2 REM * -------   ----------------- *
OT 3 REM * Draws a simple graph. You   *
JS 4 REM * specify the base temperature*
UR 5 REM * as well as the sample rate  *
QK 6 REM * (in seconds). A quiet click *
TZ 7 REM * indicates temperature read, *
CL 8 REM * a loud one means plot error.*
EQ 9 REM ******************************
ZF 10 DIM T$(5)
FR 20 T$="*****":FIRST=1
VN 40 FOR I=1536 TO 1536+158
NH 50 READ X:POKE I,X
IU 60 NEXT I
RJ 70 PRINT "Sample interval (seconds)...
   .";:INPUT RATE
RV 80 GRAPHICS 8:COLOR 1:X=0:TRAP 10000
UC 90 PRINT "Base temperature...";:INPUT
   BASE:PRINT
QM 100 REM
YL 110 FOR C=1 TO RATE
IC 120 Z=USR(1536,ADR(T$)):T=VAL(T$)
IL 130 SOUND 1,99,10,15:SOUND 1,0,0,0
DR 140 NEXT C
DV 150 POKE 77,0:REM Suppress attract.
AY 160 T=T-BASE:Y=160-(T*10)
XM 170 X=X+1
YG 180 ? " Temp=";T$;"  X=";X,"Y=";Y:PRIN
   T CHR$(28);
NA 190 IF FIRST THEN PLOT X,Y:FIRST=0
PX 200 DRAWTO X,Y
LL 220 GOTO 100
SN 999 REM
YB 10000 REM Error trap for 'Drawto'.
KY 10010 SOUND 1,99,10,15:FOR I=1 TO 20:N
   EXT I:SOUND 1,0,0,0
LX 10020 TRAP 10000:GOTO 100
EF 10099 REM
AS 32750 REM ....Themometer I/F Code....
ME 32751 DATA 104,104,133,213,104,133,212
   ,169,139,141,16,2,169,6,141,17,2,160,0
   ,132,214,173,16,208,208,251,120,165
QD 32752 DATA 16,9,1,133,16,141,14,210,16
   9,25,141,0,210,141,9,210,88,173,0,211,
   41,1,240,4,169,32,208,2,169,45,145,212
YQ 32753 DATA 165,214,240,252,41,1,208,20
   5,173,16,208,240,251,200,162,4,142,29,
   208,169,0,133,215,173,16,208,208,251
GP 32754 DATA 169,0,141,29,208,173,16,208
   ,240,251,169,4,141,29,208,173,0,211,73
   ,1,74,38,215,202,208,225,165,215,24
KU 32755 DATA 105,48,145,212,200,192,4,20
   8,204,145,212,136,169,46,145,212,169,0
   ,141,29,208,96,120,165,16,41,254,133
LZ 32756 DATA 16,141,14,210,173,16,208,9,
   4,133,214,104,88,64
```

Listing 3.

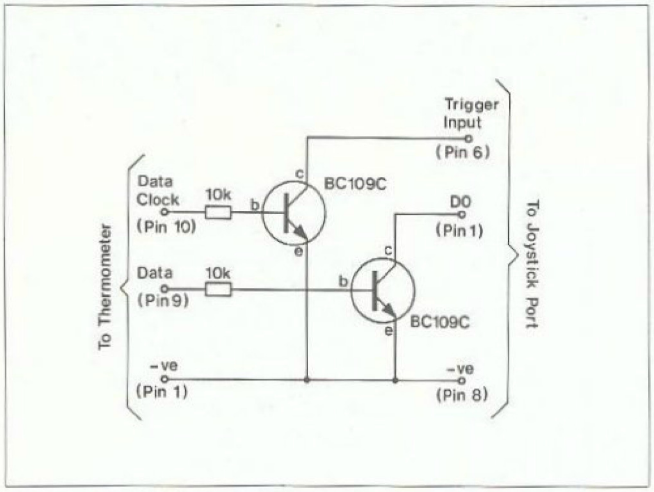
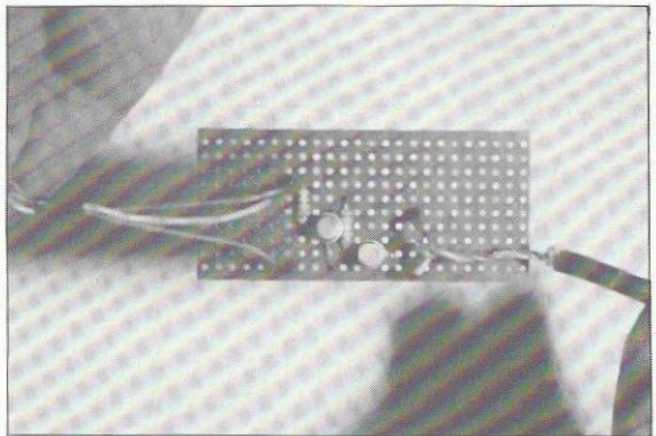


Diagram 4. Interface circuit.



Veroboard interface.

Electrical Interface

Linking the output of the thermometer is a doddle. The only difficulty I had was that the output of the clock was too feeble to drive the computer's joystick socket directly. The output is only just over a volt (inevitable really, since the thermometer is powered by a one and a half volt battery!), and does not have a high enough current sinking ability either.

This problem was solved by using a couple of transistors, one for each line. Look at Diagram 4 and you will see what I mean. This circuit will of course invert each line, but that was quite easy to allow for in the machine code subroutine. I built my circuit on a small piece of veroboard, and you can see how I have laid this out in the photograph. If you follow this layout you will not have to cut any of the veroboards tracks.

The two resistors are 10k ohms in value, and the two transistors are BC109's (in fact, virtually any NPN transistor would suffice, these just happened to be in my bits and piece's box).

Building the Interface Circuit

Take care not to leave your soldering iron on the transistor leads for too long, as they can suffer fatally if overheated!

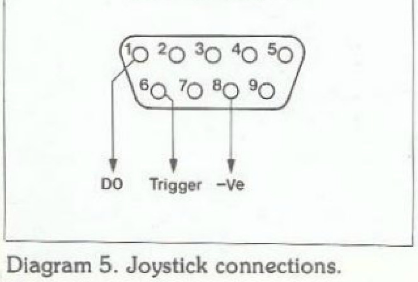


Diagram 5. Joystick connections.

When you attach your leads to the veroboard use veropins rather than soldering them directly to the board, and then shrink an inch or so of heatshrink sleeving over the join. This will help prevent the cables fraying and breaking off at the join.

It does not matter which way around the two resistors are inserted, but be sure that you place the transistors in correctly - the tag on their rim is nearest the emitter lead.

To connect this board to your computer you need to construct a 9 way plug and lead, and connect it to joystick port 1 as shown in Diagram 5. The plug is available from Maplin Electronics as a bare connector and a housing is sold separately. An alternative to this is to sabotage an old joystick (the one your dog chewed to a pulp!) and use the moulded lead. You will have to find out which leads to use, but this is quite easily accomplished using a voltmeter or a torch bulb and battery.

Connecting your board to the thermometer is even simpler, since you can solder the wires directly onto its edge connector. If you want to use the thermometer for other projects you can avoid having to solder directly to it's edge connector by using a 0.1 inch edge connector. If you do this you might have to cut away a millimeter or so of plastic from the battery holders. I found I had to do this because my edge connector was just a little too big (in thickness, not length).

Using the Options

Don't forget that you might want to select some of the extra functions available, either permanently or at different occasions. If you decide to set a function permanently you can do this by simply soldering a link between the relevant pins on the edge connector or the module's pcb directly. A better idea than this is to use a D.I.P. switch (a pcb mounted, 0.1 inch switch cluster) mounted on (and connected to) your pcb ,or, as I did in my test set-up, soldered piggy-back onto the 0.1 inch edge connector! You can see how I have done this in the photographs.

A far neater arrangement would be to incorporate both the temperature module and the little interface circuit into a small plastic box. You could then have proper switches mounted on the box. If you do this, make sure you fit a socket of some kind (a 2.5mm socket would be ideal) so that you can use an external temperature probe, something you will find essential if you want to measure the temperature of liquids, rather than just air temperature.

Using the Thermometer

So far I have assumed that you will be using your thermometer in centigrade, and consequently the temperature decoding programs interpret the first bit (D1) as the 'sign'. A high on this bit would give a negative symbol in the first character of T$ instead of the usual space. To use the thermometer in fahrenheit, however, you need to do just two things. The first is to link (or switch) contact 14 to contact 16, and the second is to add a line to your program (immediately following the USR call) to look at the first character of T$ and if it is a minus sign, change it to a '1'. For example:

```
IF T$(1,1) = "-" THEN T$(1,1) = "1"
```

This gives you the hundreds value. The other way of obtaining fahrenheit values is to calculate its value from the centigrade reading, but with this method you will not benefit from the greater resolution that you would get with the unit switched to fahrenheit (Where each tenth of a degree represents a smaller change in temperature). In fact, to obtain a good resolution with centigrade results it might even pay to operate the module in fahrenheit and calculate the centigrade equivalent!

Finally

The order code of the module from Maplin is FE33L and it costs £5.95. I hope you enjoy building this project as much as I did.

COMPUTERHOUSE

THE 1050 SUPER ARCHIVER II

ATARI

£79.95

The 1050 SUPER ARCHIVER obsoletes ALL DISK DUPLICATING DEVICES currently available for the ATARI 1050. It ELIMINATES the need for patches, and PDB files etc. Copies are EXACT DUPLICATES of the original and will run on any drive. Without exaggeration the S U P E R A R C H I V E R is the most powerful PROGRAMMING/DISK DUPLICATING device available for the 1050!!! Installation consists of a plug-in chip and six simple solder connections.....

FEATURES INCLUDE:

- BUILT-IN U.S. DOUBLER
- TRUE DOUBLE DENSITY
- SUPPORTS EXTRA MEMORY
- TOGGLE HEX/DEC DISPLAY
- SECTOR or TRACK TRACING
- AUTOMATIC DIAGNOSTICS
- DISPLAYS HIDDEN PROTECTION
- ADJUSTABLE/CUSTOM SKEWING
- THE ONLY SYSTEM THAT COPIES, CREATES PHANTOM/FUZZY SECTORS
- 810/HAPPY ARCHIVER COMPATABLE
- PRINTER SCREEN DUMP FACILITY
- BUILT-IN EDITOR - read, write ,edit/35 sectors per track
- AUTO-SPEED COMPENSATION
- HARDWARE/SOFTWARE EXPANDABLE
- BUILT-IN CUSTOM FORMATTER up to 40 sectors per track
- BUILT-IN TRACK MAPPER -up to 42 sectors per track
- DISPLAYS/COPIES D/D HEADERS
- AUTOMATIC FORMAT LENGTH CORRECTION SYSTEM
- BUILT-IN DISASSEMBLER

XL £59.95 THE CONTROLLER CARD P.P.B. II® XE £64.95

BREAKAWAY FROM YOUR ATARI'S CONVENTIONAL COMPUTING ENVIROMENT WITH THE CONTROLLER CARD PROGRAMMABLE PERSONALITY BOARD, AN INTERNAL HARDWARE ADD-ON SPECIFICALLY DEVELOPED FOR THE ATARI 800XL/XE COMPUTERS. The Controller Card allows you to add, from an impresive and expanding library, utilities/enhancements that reside in memory above commercial /home produced software. Activating any of these utilities will temporarily freeze the main program until the utility has completed it's task ,or act as an in-line enhancement, and to impress you even further, many of the utilities/program enhancements are currently not available on any other business or home computer system!!!

CUPID INTERNAL INTERFACE

'WHAT YOUR ATARI XL/XE SHOULD HAVE COME WITH AS STANDARD' Printer connection is via a standard 25-way 'D' type connector, leaving your SIO lines free. Data transfer is much faster than any SIO to parallel printer interface. Supplied with our latest SHQ™ print SYSTEM enabling you to produce "PRESENTATION QUALITY" graphics output to an Epson and compatable dot matrix printer from graphic packages such as PRINTSHOP (Broderbound),TYPESETTER SYSTEM (Xlent) etc......

CUPID KIT for the Controller Card--£24.95

NO OTHER COMPUTER SYSTEM HAS THIS FACILITY

An installation service is available for all our products on request

14 Romily Court, Landridge Road, Fulham, London SW6 4LL. Tel: 01-731 1276
Partnership: J.L. May, C. Lakey

SEARCHER

Program in TURBO BASIC

For XL/XE only
By Earl Davidson

SEARCHER is a Turbo Basic utility program allowing search and replace functions while in Turbo Basic. Turbo Basic is available from the 8 bit library. Type in the program, use the checking codes to assist you if necessary (you will require a checksum program such as 'Page 6' magazines Typo, or our own Keyo which is also in the library). Once you have finished typing, save the program to disk by using the 'LIST' command:

LIST "D:SEARCHER.LTB"

USING SEARCHER

To use the program, load Turbo Basic and type the following commands:

ENTER "D:SEARCHER.LTB"
GO # SEARCHER

SEARCHER will prompt you for the following information:

CONFIRM (Y/N)?
MATCH CASE (Y/N)?
STARTING LINE NUMBER?
ENDING LINE NUMBER?
SEARCH FOR?
REPLACE WITH?

Each line selected will be displayed. If the 'search for' string is found it will be replaced with the 'replace with' string. If you answer 'Yes' to the CONFIRM prompt, each time the 'search for' string is found the program will pause, blink the string, and ask 'Replace here? (Y/N/A/Q)?'. Respond for Yes, No, Abort, or Quit.

PROGRAM NOTES

Defaults are provided for the first four prompts above. Press RETURN to accept the default or edit the line as you wish. After the search is complete, SEARCHER returns you to the 'SEARCH FOR?' prompt. To change any of the other prompts, press RETURN on a blank line for the 'SEARCH FOR?' prompt. The program uses lines 31000 to 31056, so if your program uses these lines, they will be replaced. SEARCHER may be renumbered, if necessary, and LISTed to disk to eliminate any line number conflicts with other utilities. All variables begin with a 'Z' so that you may identify those which belong to this utility. These variables may be cleared out of your program by using the following commands:

LIST "D:PROGRAM.LTB",0,30999
NEW
ENTER "D:PROGRAM.LTB"
SAVE "D:PROGRAM.STB"

If you have any long lines that were typed in with statement abbreviations, or if you use a 'replace with' string longer than the 'search for' string, you may have problems when the line is re-entered. Care should be taken to avoid this. I suggest you use the CONFIRM option until you are familiar with the program. Sometimes the 'search for' string is found imbedded in unusual places! The MATCH CASE option allows you to determine if upper case, lower case and inverse is significant in the search. 'Yes' requires an EXACT match. 'No' will allow a search for 'A' to find 'A', 'a' and the inverse of both. When MATCH CASE is 'No' the Turbo Basic command 'UINSTR' is used. Apparently there is a bug in this routine. Searching for '1' (decimal 49) will also find 'ctrl Q' (decimal 17) and its inverse (decimal 145). Searching for '2' will also find 'ctrl R' (decimal 18) and its inverse (decimal 146). This is an offset of 32 and appears to hold true for all characters from decimal 33 to 64. If you use SEARCHER with the CONFIRM option off, MATCH CASE off, and search for one character at a time, you may damage any machine language routines you have imbedded in strings. Use these options carefully! I hope you find SEARCHER to be a useful utility!

Note: Anything underlined in this program should be typed in inverse.

```
HQ 31000 ------------------------------
YF 31001 REM :        SEARCHER         :
CZ 31002 REM :    for Turbo BASIC XL   :
AI 31003 REM :     by Earl Davidson    :
TM 31004 REM :   Version 1.0 (3/87)    :
IP 31005 ------------------------------
VQ 31006 # SEARCHER:ZSCRMEM=DPEEK(88)
BY 31007 GRAPHICS %0:TRAP 31011:POKE 709,
   12:POKE 710,%0:POKE 712,%2:POKE 752,%0
   :DPOKE 729,280:ZLM=%1:POKE 82,ZLM
DE 31008 DIM ZLINE$(200),ZNEWLINE$(200),Z
   SRCH$(200),ZRPLCE$(200),ZCNFRM$(10),ZC
   ASE$(10),ZBLNK$(840),ZINVERT$(41)
DT 31009 ZBLNK$(840)="":FOR I=1 TO 40:RE
   AD D:ZINVERT$(I,I)=CHR$(D):NEXT I
QL 31010 ZCNFRM$="Yes":ZCASE$="Yes":ZSTAR
   T=0:ZEND=31000
XB 31011 TRAP 31011:POSITION ZLM,%0:? "
    CONFIRM: ";ZCNFRM$:POSITION ZLM+11,%0
   :INPUT " ",ZCNFRM$
ZZ 31012 ZCNFRM$=ZCNFRM$(%1,%1):IF ZCNFRM
   $="N" OR ZCNFRM$="n":ZCNFRM=%0:ZCNFRM$
   ="No":ELSE :ZCNFRM$="Yes":ZCNFRM=%1:EN
   DIF
ZP 31013 POSITION ZLM+12,%0:? ZCNFRM$;"
         "
TN 31014 TRAP 31014:POSITION ZLM,%1:? "MA
   TCH CASE: ";ZCASE$:POSITION ZLM+11,%1:
   INPUT " ",ZCASE$
IR 31015 ZCASE$=ZCASE$(%1,%1):IF ZCASE$="
   N" OR ZCASE$="n":ZCASE=%0:ZCASE$="No":
   ELSE :ZCASE$="Yes":ZCASE=%1:ENDIF
ZH 31016 POSITION ZLM+12,%1:? ZCASE$;"
         "
WF 31017 TRAP 31017:POSITION ZLM+17,%0:?
   "STARTING LINE: ";ZSTART1:POSITION ZLM
   +31,%0:INPUT " ",ZSTART1:ZSTART=ZSTART
   1
QO 31018 TRAP 31018:POSITION ZLM+17,%1:?
   "  ENDING LINE: ";ZEND:POSITION ZLM+31
   ,%1:INPUT " ",ZEND
CN 31019 IF ZEND<ZSTART THEN ? "";:GOTO 3
   1017
VI 31020 MOVE (ADR(ZBLNK$)),ZSCRMEM+121,8
   40
IC 31021 POSITION ZLM,%3:INPUT "  SEARCH
   FOR: ",ZSRCH$:IF NOT LEN(ZSRCH$) THEN
   GO# SEARCHER
XV 31022 INPUT "REPLACE WITH: ",ZRPLCE$:I
   F NOT LEN(ZRPLCE$) THEN GO# SEARCHER
PD 31023 TRAP 40000:ZCNT=%0:ZCHNGD=%0:POK
   E 752,%1
AZ 31024 DO
BX 31025   ZSTTBL=DPEEK(136):ZLINE=DPEEK(
   ZSTTBL)
SZ 31026   WHILE ZLINE<ZSTART:ZSTTBL=ZSTT
   BL+PEEK(ZSTTBL+%2):ZLINE=DPEEK(ZSTTBL)
   :WEND :IF ZLINE>ZEND THEN POP :GOTO 31
   055
ZG 31027   POSITION ZLM,11:LIST ZLINE:POS
   ITION ZLM-%1,12:POKE 842,13:INPUT " ",
   ZLINE$:POKE 842,12:ZS=%0:ZF=%1
OC 31028   WHILE ZF
GP 31029     IF ZCASE:ZF=INSTR(ZLINE$,ZSR
   CH$,ZS):ELSE :ZF=UINSTR(ZLINE$,ZSRCH$,
   ZS):ENDIF
```

```
LQ 31030      IF ZF
UQ 31031         ZCNT=ZCNT+%1:ZCHG=%1:POSIT
   ION ZLM,11:LIST ZLINE
BH 31032         IF ZCNFRM:POKE 764,255:POS
   ITION ZLM,23:? "REPLACE HERE (Y/N/A/Q)
   ? ";:ZJ=ZSCRMEM+400+ZF
VV 31033         ZJ=ZJ+(ZLM*(ZF>80-(ZLM*%
   2)))+(ZLM*(ZF>40-ZLM)):ZL=LEN(ZSRCH$)
VB 31034         IF ZF<40 AND ZF+LEN(ZSRC
   H$)>40 THEN ZL=ZL+1
ZM 31035         IF ZF<79 AND ZF+LEN(ZSRC
   H$)>79 THEN ZL=ZL+1
YM 31036         REPEAT :ZA=USR(ADR(ZINVE
   RT$),ZJ,ZL):PAUSE 10:UNTIL PEEK(764)<>
   255
IT 31037         GET ZKEY:PUT ZKEY:POSITI
   ON ZLM,11:LIST ZLINE:IF ZKEY=89 OR ZKE
   Y=121:ZCHG=%1:ELSE :ZCHG=%0:ENDIF
QS 31038         IF ZKEY=65 OR ZKEY=97 TH
   EN POP :POP :POP :POKE 752,0:CLS :GO#
   SEARCHER
FZ 31039         IF ZKEY=81 OR ZKEY=113 T
   HEN POP :POP :POP :POKE 752,0:END
AM 31040      ENDIF
LS 31041      IF ZCHG:ZCHNGD=ZCHNGD+%1:Z
   NEWLINE$=ZLINE$(%1,ZF):ZNEWLINE$(LEN(Z
   NEWLINE$))=ZRPLCE$
EA 31042         IF ZF+LEN(ZSRCH$)>LEN(ZL
   INE$) THEN GOTO 31044
CA 31043         ZNEWLINE$(LEN(ZNEWLINE$)
   +%1)=ZLINE$(ZF+LEN(ZSRCH$))
IM 31044         ZLINE$=ZNEWLINE$:MOVE (A
   DR(ZBLNK$)),ZSCRMEM+400,560
EA 31045         POSITION ZLM,12:FOR ZI=%
   1 TO LEN(ZLINE$):? ";ZLINE$(ZI,ZI);:
   NEXT ZI
VH 31046         POSITION ZLM,19:? "CONT"
   :POSITION ZLM,9:POKE 842,13:STOP
RF 31047         POKE 842,12
CA 31048      ENDIF
XZ 31049      ZNEWLINE$="":ZS=ZF:ZS=ZS+(
   (ZCHG)*(LEN(ZRPLCE$)-%1)):IF ZS>LEN(ZL
   INE$) THEN ZF=%0
WU 31050      ENDIF
YL 31051      MOVE (ADR(ZBLNK$)),ZSCRMEM+4
   00,560
TY 31052   WEND
GI 31053   ZSTART=ZLINE+%1:ZLINE$=""
JS 31054 LOOP
NZ 31055 POKE 752,%0:POSITION ZLM,19:? ZS
   RCH$;" was found ";ZCNT;" times.":? "I
   t was changed ";ZCHNGD;" times."
CI 31056 ? CHR$(29);" PRESS ANY KEY TO CO
   NTINUE";:GET ZKEY:ZSTART=ZSTART1:GOTO
   31020
CD 31057 DATA 104,208,1,96,201,2,240,8,17
   0,104
SE 31058 DATA 104,202,208,251,240,243,104
   ,133,213,104
LG 31059 DATA 133,212,104,104,56,233,1,41
   ,127,168
WT 31060 DATA 177,212,73,128,145,212,136,
   16,247,96
```

256K UPGRADE

FOR • YOUR • 800XL

by BUG

Frontier Software of Harrogate are currently marketing the 'Rambo' board to upgrade your 800XL to 256K memory, although you have to provide the DRAM chips and your own DOS, and the cost is around 30 pounds. However the original details of the Rambo XL upgrade were published in the Public Domain in the MAAUG newsletter in an article written by Mike Redmond. A considerable amount of credit must also go to Claus Buchholz whose article published in the October edition of BYTE magazine inspired the first DIY modifications.

It should be pointed out that not all 800XL's are suitable for modification. The Rambo upgrade works best with those machines fitted with ANTIC chip part number C021697 or C021698. The ANTIC chip is designated U7 on the 800XL printed circuit board. Rambo has been tried in other machines and the only problem encountered on some of the earlier machines is a delay in excess of 10 seconds when you try to coldstart and re-boot another program. There are a few 800XL's which are incompatible with this upgrade, they are the ones made in Malaysia. These machines only have 25 chips inside instead of the normal 29! If your machine has only 25 chips, you are advised to use the type of upgrade that plugs into the rear port and does not require any internal modification.

The Rambo XL upgrade attempts to maintain compatibility with the 128K 130XE, and the compatibility is pretty good, that is unless your software makes specific use of the 130XE's ability to bank switch the video RAM. Rambo XL cannot cope with that, otherwise it has been found that RAMDISK.COM works without modification, as does DOS 2.5. To take full advantage of the 256K upgrade the standard handler software must be modified, a fully modified version is available from the Monitor 8 Bit Library. This DOS allows you to define two full size (707 sectors) single density RAMdisks as D7: and D8:.

How does it work?

Basically the 64K RAM chips originally installed in the 800XL are removed and replaced with 256K RAMs. Then a new address decoder circuit is installed in place of the chip that originally did the address switching, see Figure 1. Lines on the PIA are used to control the address selection in such a way that simulates and extends the 130XE use of these lines. Instead of switching memory in 32K blocks, as done in the BYTE design, the Rambo XL switches memory in 16K blocks, like the 130XE. This not only allows the Rambo XL to work like the 130XE, it also removes the problem associated with swapping out operating system work space in the lower 16K of RAM. Rambo XL switches the second 16K bank bracketed by the hex address 4000 to 7FFF.

The 256K RAM space is divided into 16 segments that can be moved into the 4000 to 7FFF space. Four of the segments are normally used to simulate standard 800XL operation, the other segments are swapped in and out under control of RAMDISK or other RAM management software. The circuit produces three address signals for the 256K RAM chips for the lower address bits, address lines A0 through A13 (locations 0000 through 3FFF). This new circuit is also de-activated (using signal XE) when the computer attempts to address locations above 8000. In the 7FFF range, sixteen new addresses can be generated on RA6, RA7 and RA8, to activate one of the segments. The new RA6 and RA7 addresses are controlled by lines PB2 and PB3 from the PIA chip. These are the same lines used on the 130XE. PB4 is also used here (to generate the XE signal) as it is used on the 130XE. To enable extended memory addressing RA8, which is not generated in the 800XL or the 130XE, is generated here by multiplexing PB5 and PB6 to the RA8 line. This provides access to the full 256K RAM by dividing it into four 64K segments that are then sub-divided in to four 16K segments using PB2 and PB3. Unfortunately, PB5 is used on the 130XE to control video bank switching, therefore when using 130XE software that expects to bank switch the video, on the Rambo XL terrible things will happen, such as a lock up!!!!

Warning

Before you start taking your machine to bits, please be sure that you know what you are doing! Monitor magazine (and the club) will not accept any responsibility for what you do to your machine in the privacy of your home or workshop, also don't forget about your warranty. Make sure your machine is out of warranty before you break into it for

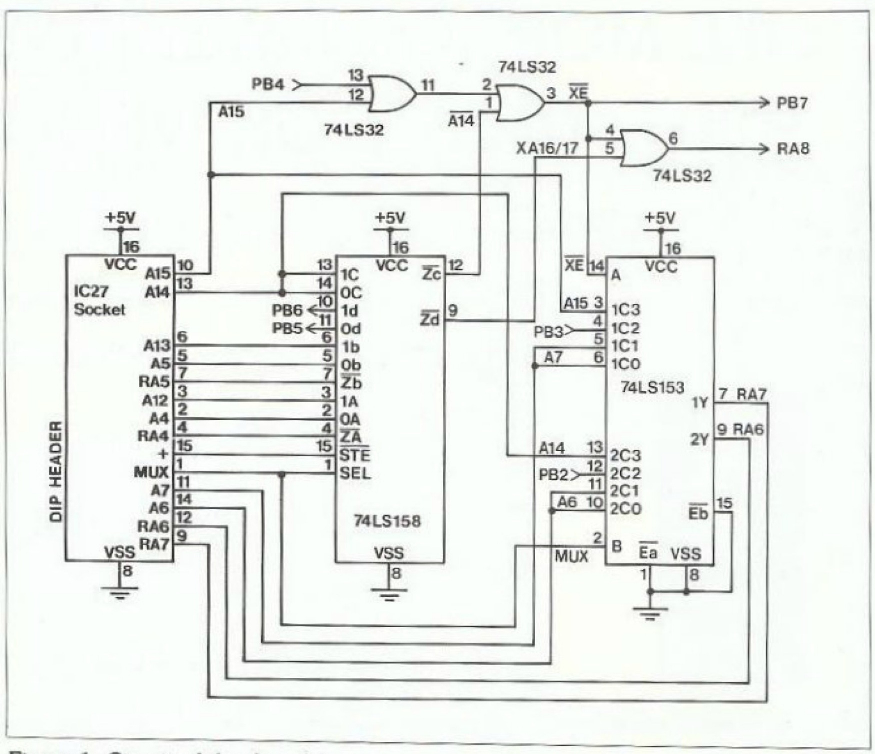
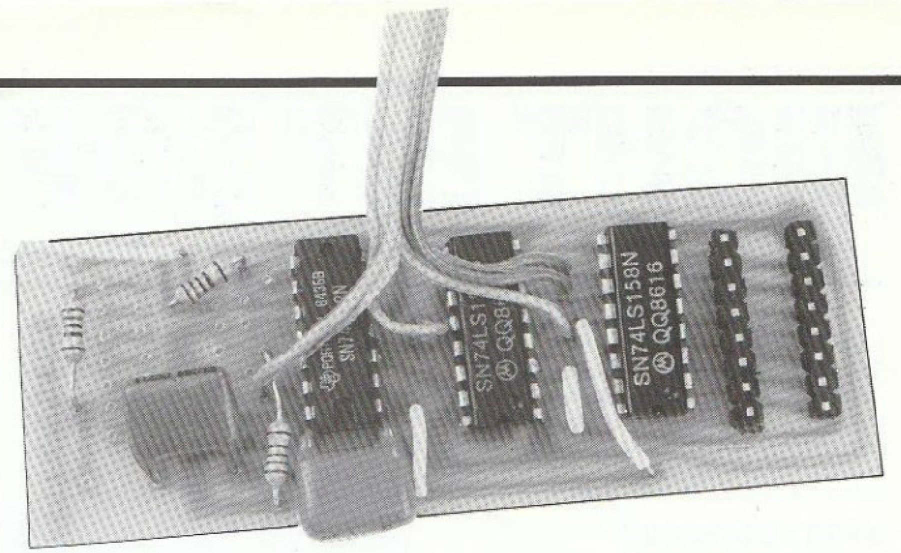


Figure 1. Circuit of the decoder.

something like this. If you are still determined to go ahead, make sure you read and follow the construction details very thoroughly before starting work. This modification has worked well for us, we hope it does the same for you!



Construction

If you wish to construct your own decoder board, the schematic is shown in Figure 1, and a list of parts is given in Table 1. The layout is not critical, especially if you plan to use a header plug and ribbon cable to connect the decoder to the main PCB inside the Atari. The alternative is to use a long leg wire wrap socket soldered through the decoder board, then plug direct onto the IC socket. If you have difficulty building the decoder, a ready built and tested one is available from Micro Discounts in Birmingham (details at end of article).

When you have completed the decoder the real work begins. Turn your machine over on its face, remove the 6 phillips head screws. Turn the machine on to its back, whilst holding both halves of the case shell. The keyboard is connected to the PCB by a ribbon cable connector, this must be removed from its socket, (do not use any tools) hold the connector on each edge between your fingers and gently ease upwards out of its socket, when removed place keyboard to one side. Remove the 2 screws holding the PCB to the bottom of the plastic shell (both are located at the rear of the RF shield). Remove the PCB from its case, it is a tight fit, but if you lift the left side front first, you may find it easier. Remove the RF shielding from the top and the bottom of the PCB, some 800XLs have bent over tabs to hold the shield together, if you have one of these straighten the tabs and gently ease the two shields apart, if you have screws just remove them and part the shields.

Now do an IC count. Your machine should contain 28 chips. Now check your ANTIC chip in socket U7, if it reads C021697 or C021698 you are in luck and can continue, any other numbers may not be compatible with this modification.

Now for the tricky bit, changing the RAM chips. If the RAMs are not socketed you will need to desolder them. Cover your work surface with tin foil, place your PCB on the foil and keep it on there, this will reduce the possibility of destroying the RAMs from a static discharge. You must remove the RAMs from positions U9 to U16 inclusive. It is recommended that you remove one IC at a time, fit a socket (if required) and insert a 256K RAM chip. When this one has been fitted, clear away all loose solder and foil. Plug the computer into the mains and power up, if Basic enables proceed to the next RAM chip, but if your computer fails to enable Basic, you'll know where to start looking for a bad solder joint or a lifted track, etc. If you haven't got a desoldering workstation (and who has?), the best way to remove ICs is to use a pair of small sharp side cutters to cut the legs off on either side of the chips, then you can remove the main body of the IC, then it is easy to remove the remains of the legs and excess solder from the holes. Carefully check that none of the tracks have been damaged.

Now remove the chip from location U27 and resistor R32 located at the top end of the RAM stack. Double check that the location nitch on all RAM chips point to the left side of the PCB when looking at your 800XL from the front. Now install the decoder in to socket U27, solder the fly lead from RA8 pad on the decoder to the right hand hole of the vacated R32. Connect the ribbon cable to the PIA chip (U23) soldering the ribbon cable direct to the following pins; PB2 to 12 (green), PB3 to 13 (yellow), PB4 to 14 (orange), PB5 to 15 (red) and PB6 to 16 (brown). That's it! Not too bad was it!!

Carry out the following test before you re-assemble the computer. Remove all tin foil from under the PCB and any loose solder. Insert all connecting cables, except the keyboard (at this stage). Again check for loose bits of solder, if all is clear power up. If you have completed all correctly, you should get the Basic 'READY' in the top left hand corner of the screen. If you do not, switch off immediately and re-trace your steps, checking solder joints and that all ICs are seated properly in their sockets.

When you have a working machine, switch off and re-assemble the RF shields on the PCB. Before you fit the top shield, put a strip of insulation tape over the top of the PIA chip covering the ribbon cable connections. The top shield is a tight fit, but the screws should pull it down to shape, make sure you do not trap any loose wires between the shield and PCB. Replace the PCB into its case, re-connect the keyboard connector, replace the keyboard and fit the case retaining screws through the bottom of the case. That's it!

Now complete the following test procedure. Boot in a standard DOS 2.5 disk. If all is OK your computer will think it's a 130XE. If you have the special patched DOS disk (available from the 8 bit library), boot it in with Basic enabled, watch the load screen, it should say 'Setting up Rambo 256K RAM disk'. Type 'DOS' followed by pressing RETURN to get to the menu, also notice the speed, it's in RAM. Type 'A', press RETURN twice, this will read the disk directory. Type 'A' and press RETURN once. Type 'D8:RETURN', this will show 620 empty sectors, also DOS and DUP. Type 'A' and press RETURN once, type 'D7:RETURN', this will show 707 empty sectors.

Now you have all that RAM available, how about some of you budding programmers coming up with some spectacular demos, i.e. digitised speech that lasts for more than 7 seconds? Try bank switching? How about a printer buffer using D7/D8? A word processor that puts all its files, i.e. spell checker or mail merge, into RAM and then switches to them with out spinning up the drives? Think of the time saved, let alone wear and tear on your drives.

Component List
RAM chip 41256 150ns (Qty 8)
74LS32 (Qty 1)
74LS153 (Qty 1)
74LS158 (Qty 1)
Resistor 33 ohms (Qty 2)
Resistor 3.3K ohms (Qty 1)
Capacitor 0.1uF (Qty 2)
Low profile IC socket (Qty 9)
Multi-strand ribbon cable (Qty 18")
Header Plug (Qty 1)

Table 1. Parts list.

Appendix

As mentioned earlier, the decoder is available ready-built and tested from Micro Discount, P.O. Box 946, Sutton Coldfield, West Midlands, B74 3EZ. The cost is £16.50 plus £1 postage or you can purchase a complete kit including decoder, RAM chips, sockets and the DOS 2.5 disk for £58 plus £2.50 postage and packing. Micro Discount also offer a fitting service at a total cost of £65 plus £4.50 for insured return postage.

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

OH DAMN!

by Les Howarth - Preston.
Place your tiles cleverly on the board and you'll win.
Runs in 48K min. Disk Only.

PONTOON II

by Chris Guise - Sidmouth.
Good version of the well known card game.
Runs in 48K min. Disk or Cassette.

Demos

PASCAL SAMPLER

A selection of Kayan Pascal demo programs, including Wordlist, Fog, Prfile and Cloze.
Runs in 48K min. Disk only.
Requires 1 side of a disk.

STEWS GALLERY I

by Stewart Bullough - Hadfield.
24 highly colourful Touch Tablet pictures.
Runs in 48K min. Disk only.
Requires 1 side of a disk.

STEWS GALLERY II

by Stewart Bullough - Hadfield.
18 Fun with Art pictures displayed one after the other.
Runs in 48K min. Disk only.
Requires 2 sides of a disk.

STEWS MOVIES

by Stewart Bullough - Hadfield.
Animated movie shorts, some a little on the risky side.
Runs in 48K min. Disk only.
Requires 1 side of a disk.

Utilities

DIRLIST

by T. Wood - Reading.
Turbo Basic program which lists disk directories in alphabetical order on an Epson printer.
Runs in any size. Disk only.
Requires Turbo Basic.
XL/XE only.

***** STAR PROGRAM *****

HACKERS TOOLKIT

Seven useful programs that are used to work on object files, clean up disks, etc. Includes a disassembler, disk sector scanner, simple assembler and a sector copier amongst others.
Runs in any size. Disk only.
Requires 2 sides of a disk.

****

INTEREST

by P. Budge - Stoke-on-Trent.
Microsoft Basic program which copes with monthly, quarterly and annual interest input. 2 versions, output to Printer or Screen.
Runs in any siz. Disk only.
Requires Microsoft Basic.

MINI 3D CAD

Nice little 3D drawing program with some excellent features.
Runs in 48K min. Disk only.
Requires 1 side of disk.

PRINT SHOP EDITOR

Create your own Print Shop pics.
Runs in 48K min. Disk only.
Requires 2 sides of a disk.
Load with Print Shop.

TURBO SEARCHER

Gives string search routine for Turbo Basic programs.
Runs in 48K min. Disk or Cassette.
XL/XE machines only.

256K UPGRADE PATCH

by BUG.
If you build the 256K upgrade, these files will be very useful.
Disk only. 1050 only.
Requires 1 side of disk.

Spartados Tool Kit

Review by Richard A. Gunter

Upgrading that trusty 8 bit system can be an attractive alternative to dropping several hundred pounds on a new computer. By adding a little hardware and jazzing up the software, we can give that old workhorse a new lease of life.

ICD Inc., of Rockford, Illinois, USA, has achieved a commanding position in this market. They're the authors of Spartados, the US Doubler, R-Time 8 cartridge, and more. Their latest offering is the Spartados Tool Kit.

A little background is needed on this powerful DOS in order to understand what's in the Tool Kit.

An important Spartados concept, familiar to users of MS-DOS or DEC mainframes, are directory hierarchies. A directory may contain directories as well as data files. When you list the contents of a directory, the files in its subordinate directories are not displayed. You must identify the directory whose contents you want.

Unlike DOS 2 and 2.5, Spartados is a command oriented operating system. With no menu, a command is typed to perform a function. Thus, 'DIR D2:' displays the directory of the disk in Drive 2.

A number of the commands are built into the initial load, while the rest are external. An external function is invoked in exactly the same way as a built-in function: just type its name. The program of that name is loaded and run.

The command philosophy makes this kind of DOS almost infinitely extendable. You can write your own functions, and think of them as extensions of the DOS.

That's exactly what the Tool Kit is: a set of additional functions for Spartados. Most of the new functions are said to work with earlier versions of Spartados, but I did my tests with version 3.2d on a 130XE. You shouldn't have any trouble with an 800XL, but the later versions of Spartados won't run on an 800. The same may be true for some of the tools.

The Tool Kit disk contains eight utilities: MIOCFG, COMMAND, RENDIR, SORTDIR, VDEL, WHEREIS, DOSMENU, and DISKRX.

MIOCFG

This is used to save and reload MIO configurations. I couldn't evaluate this one, since I don't have an MIO. According to the manual, MIO users without a hard drive have no way to save configuration data without this utility.

COMMAND

This program allows you to set up characters to be sent to the DOS with a single keystroke. COMMAND allows more key definitions (20 including Control 1, 2 and 3), and defines some commands not supported before.

The new built-in commands allow you to change screen colour, force a re-boot by typing 'cold', and change the prompt to show the directory path (nifty!).

One of the neatest new commands provided by COMMAND is 'ibm', which gives you a 'last line recall'. with some bells and whistles. The most elaborate recall I've seen is on the VAX systems, where you could recall up to 20 command lines. Here you get one, but it's really handy.

The Tool Kit disk contains 'BAT' files for Action!, Mac65 and Basic programmers. My main gripe is that COMMAND provides no way to list your current key definitions. Since it intercepts keystrokes and does things to colour registers, this program may interfere with some programs.

RENDIR and SORTDIR

These are directory manipulation functions. RENDIR permits you to rename a directory. Without this little gem, you'd have to delete all the files in the directory to be renamed, delete the directory, create a new directory with the new name, then copy the files into it. A real hassle. SORTDIR sorts the current directory. The options are to sort by size, file name, extension or date, in either ascending or descending order. Some of the sorts have a secondary key. For example, sort by date will use time as the secondary key. You don't have complete control of the sort, but the options seem to be entirely adequate.

VDEL

This program, unlike the normal Spartados ERASE command which is unforgiving of user errors (files selected are immediately erased), provides that second chance. You are prompted to erase each file. I, of course, never make a mistake with ERASE...

WHEREIS

This is another useful little rascal, if you use multiple directories. It searches for a file name and reports the path needed to reach it. Since WHEREIS also allows wildcards in the file spec, it can be used to get much the same results as TREE (supplied with Spartados). Thus, the following command will find all the Basic files on a disk:

WHEREIS *.BAS

DOSMENU

This provides a menu resembling those of DOS 2 and 2.5, and sharing some of the same option codes. 'C' is a file copy, and so on. The manual thoughtfully points out that DOSMENU eats some RAM, and may not be compatible with some programs. It requires about 2.6K.

You can invoke this program from command mode by typing DOSMENU, or run it automatically by renaming a copy of it as AUTORUN.SYS. One annoying drawback is that, once activated, there is no escape other than to re-boot (option Z). It doesn't seem to totally replace the command processor; rather many of its options generate the equivalent command line, which appears momentarily.

This is the only Tool Kit program on which I have noticed an apparent bug. On my 130XE, booted with Basic active, option 'B' (run cartridge) transfers the control to Basic, as expected. If I do anything in Basic, the computer locks up. For example, I booted as above, entered Basic and then typed the immediate statement : 'HELLO'. This should display the word HELLO on the screen. However, output is garbled, and the computer locks up.

The problem can be avoided by booting with Basic off, then turning it on via the built-in command 'BASIC ON', or by booting with Basic on and toggling it, first off then on.

DISKRX

This program is by far the largest and most complex of the new functions. You may never need it, but if you need it, you really need it! DISKRX is a disk editor and damage control program. Its file functions are for Spartados files; no series support is available for Atari format files and directories. The program is density smart, and 256 byte sectors are displayed on two screens. Navigation between the screens is simple and convenient.

In some respects, DISKRX reminds me of the old DISKEY program for single density Atari format disks. There's quite a bit of power here, but you have to know what you're doing. A working knowledge of Sparta disk format is essential.

Manuals

This brings us to the manuals. The Tool Kit manual is just 14 pages and doesn't have the finished look of other ICD manuals, but it does contain the information you'll need to use the Tool Kit. Concerning the disk format information (needed for DISKRX), the Spartados manual is somewhat sketchy. It took me a few minutes of playing with DISKRX to figure out how everything hangs together, and I'm no beginner.

I've heard that ICD is planning to release a new version of Spartados (a cartridge) some time in the future. A bank switching cartridge should accommodate a lot of functions that are external in the current versions of the DOS. Will the Tool Kit be built into the cartridge? Dunno. It wouldn't surprise me to see some of the smaller functions from the Tool Kit migrate to the cartridge. Larger and less frequently used items, like DISKRX, probably won't.

Finally

In summary, the Spartados Tool Kit is a worthwhile addition to a Sparta-oriented system. DISKRX alone should be worth the price. The other functions, added to the impressive set supplied with the DOS, provide a lot of niceties that aren't available elsewhere. Big computer stuff on the 8 bit!

Reprinted from November 87 Current Notes.

The Expander

Review by John Pilge

The Expander is designed for the Atari XL/XE computers with a disk drive and memory expanded to 256K or 512K. It allows you to section memory as several double density or single density RAMdisks. Even if you are using a DOS that doesn't support a RAMdisk (Atari DOS 2.5, SmartDOS, etc.), Expander fools the DOS into reading from RAMdisks. You can even configure a RAMdisk as drive 1 and boot from RAMdisk. You can also copy files and format disks without going to DOS.

Expander also acts as a translator for older third-party software (Serpintine, Protector II, etc.). It has extra commands to force a coldstart, break out of a program (useful in using RAMdisks with commercial software such as word processors), and one-button save and load commands. Expander increases the keyboard response rate so the cursor moves faster.

In testing, Expander worked successfully with all Basic languages. It is therefore the perfect OS for Turbo Basic.

Expander has only a few drawbacks. If a commercial program uses function keys, you may not be able to jump out of the program to use some of the Expander commands. Electronic Arts programs do not run with ANY custom OS, including Expander. If your memory expansion is made by Newell Systems, you cannot boot from a RAMdisk or execute a binary file from a RAMdisk. Newell uses non-standard techniques for extra RAM.

Installing the Expander chip is simple on the 800XL, just replace the OS chip in your computer. With the XE computers you will find the chips are soldered in place, so you will have to desolder the chip, solder in a socket, and then put in the Expander. Newell Systems (3340 Nottingham Lane, Plano, Texas, U.S.A.) make a device called the Switchboard that allows you to have up to three Operating Systems, you select them by using a switch that is on the 'outside' of your computer. Users with a memory expanded XL/XE will find Expander handy for programming, ideal for making use of your extra memory, and great for saving your disk drive motors.

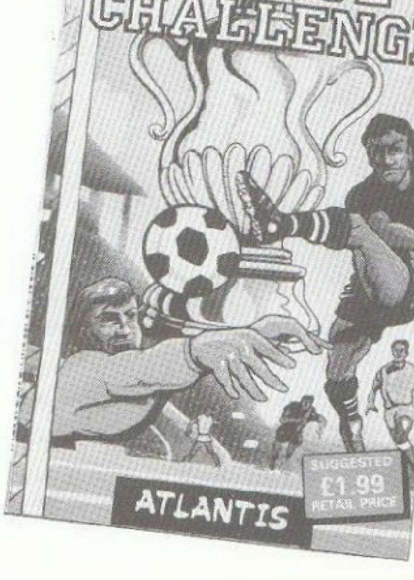
Expander is available from Synergy Concepts, Box 421370, Sacremento, CA 95841, U.S.A. Phone: (916) 922 4119. Price is $49.95.

League Challenge

From Atlantis
Cassette, £1.99
Review by S. Oker

League Challenge is a quality football league simulation streets above the usual 'budget' fodder we've come to know and detest. In this 'game' you not only are a footballer, but you get to be manager too. You can select your team from one of 64 listed (or change the name to one you like, I'm usually Goalgetters United). As you have probably guessed you start at the bottom of division 4. Your team will have to play 15 matches in the division and try to gain promotion. There are also 'knock-out' cup games to play.

For each match you can select your players from the 15 players in your



squad, some players have better qualities than others, but some are fitter. You must decide! Watch it though, some players may be injured and unfit to play, you may decide that a certain player is just not up to your standard, in which case you can sell him on the transfer market.

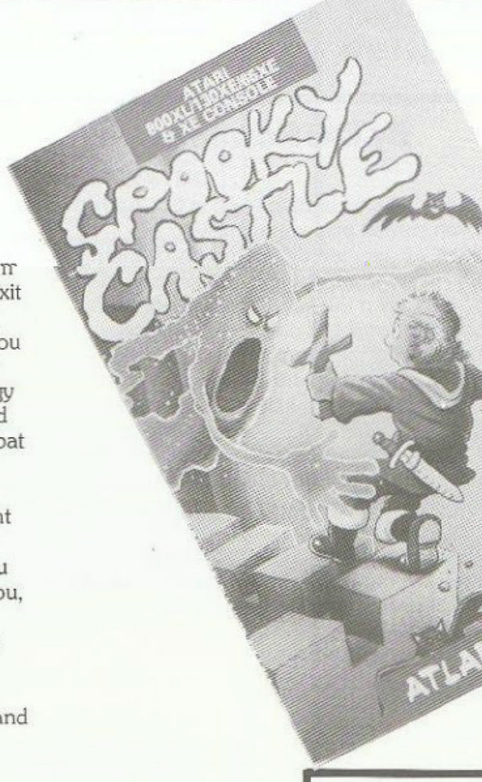
After a match, the results from other games played that day will be posted, and the new league positions shown with full details of goals for and against, points earned and total matches played. Club finances are also given so you can see if you're actually making any money or not.

All in all a darn good game, worth every penny, in fact I would gladly pay more for this if necessary. Great one Atlantis, keep it up.

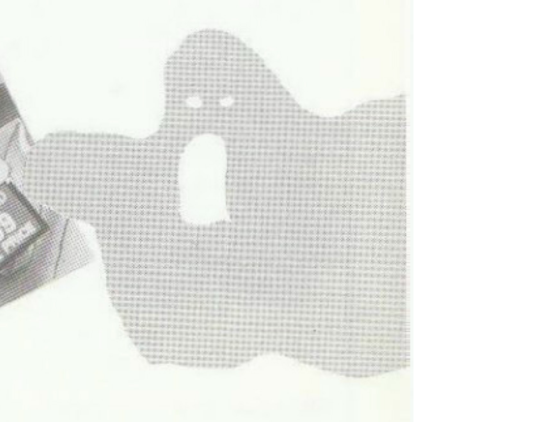
Spooky Castle

From Atlantis
Cassette, Price £1.99
Review by Tommy Taylor

Spooky Castle is a 17 level platform type arcade game. Each level has an exit for which you must locate the key to enable you to pass to the next level. You are given five lives to accomplish your task and you only have so much energy for each level, and this is being drained from your body every time a vampire bat passes over you. Not only that, but ghosts keep appearing out of the woodwork and fly helter skelter straight at you! You have no defence, you can only avoid them. If a ghost catches you and wraps its deadly shroud around you, you're a gonna!



The graphics on this game are not bad (for a budget game they are very good), the sound effects are OK too. Movement of your man is by joystick and pressing fire makes him jump, and boy does he need to jump! The game-play is fast and furious and at times frustrating, but most of all its addictive in the extreme. I think Atlantis have another hit on their hands.



OH DAMN!

By Les Howarth

Here's a great little game for you to type in. The two reference codes at the start of each line are checksum info for use with either Typo (from Page 6) or our own Keyo (available from the library). Their are quite a few control characters in this listing so type carefully. If you have any problems, the program is available from the library.

The object of the game is to score the most points out of 4 games. On starting you will be dealt 4 random tokens. These may be all different or some may be duplicated. The Atari is also dealt 4 tokens but you don't see which until the end of each game.

Each of these symbols can be placed on a 4 * 4 grid until the grid is full. Any token can be played any number of times, i.e. you are dealt 1 parrot, 2 keys, and 1 man; you could if you wanted, just keep playing the keys. Use a joystick to select a symbol to play, by placing your cursor over the symbol and then pressing the trigger. Move to the position you want to put your token in and press the trigger to deposit it.

Moves alternate between you and the Atari. Points are scored by placing tokens in patterns as shown at the bottom of the screen. In order to score for (say) the corners you would have to have one of each of your tokens in each corner. In the above example you would have to have a parrot, 2 keys and a man on the corners. The ORDER of the tokens DOES NOT MATTER. If you had an across line then you would score with: Parrot, Key, Man, Key or Key, Parrot, Man, Key, or any other combination.

During play you may be able to deduce which tokens the Atari is holding, and may be stop it scoring by placing tokens in blocking moves. When the board is full, the score for that round is calculated. Four more tokens are dealt and this continues for 4 games with you starting the first game then alternating the start with the Atari.

That's about it, get typing, I'm sure you'll enjoy the game.

Listing 1.

```
BG 10 PRINT "Please wait while the data"
LX 20 PRINT "file is being written..."
PY 30 OPEN #1,8,0,"D:FONT"
PV 40 FOR I=1 TO 503
SV 50 READ D:PUT #1,D
IU 60 NEXT I
LN 70 CLOSE #1
VE 1000 DATA 1,16,126,51,22,97,97,22
DJ 1010 DATA 22,97,51,51,83,83,83,83
OM 1020 DATA 83,131,83,131,85,133,85,53
RS 1030 DATA 83,83,85,85,85,53,85,85
MA 1040 DATA 85,85,3,51,4,64,68,68
CC 1050 DATA 64,4,64,4,64,4,68,68
TX 1060 DATA 68,64,4,64,4,64,4,64
RU 1070 DATA 4,64,4,68,4,68,4,64
WJ 1080 DATA 4,64,4,68,4,68,4,68
IQ 1090 DATA 34,34,56,56,56,56,56,56
SK 1100 DATA 56,56,131,131,131,131,131,13
    1
VB 1110 DATA 56,56,56,56,56,56,131,131
ED 1120 DATA 131,131,131,131,56,56,56,56
WY 1130 DATA 56,56,34,34,1,32,1,0
SQ 1140 DATA 1,32,132,0,132,32,1,0
FJ 1150 DATA 129,32,129,24,1,16,0,18
UH 1160 DATA 0,16,0,18,0,16,0,18
YT 1170 DATA 1,16,1,18,1,16,7,23
XJ 1180 DATA 68,68,68,116,71,119,71,119
DW 1190 DATA 71,71,71,71,68,71,68,71
VQ 1200 DATA 68,119,68,119,68,116,68,116
EN 1210 DATA 68,116,68,68,68,116,68,116
JM 1220 DATA 68,116,68,68,6,102,6,6
HI 1230 DATA 6,6,6,6,6,6,6,6
VK 1240 DATA 6,102,0,96,6,96,0,96
ME 1250 DATA 0,96,0,96,0,96,6,102
QP 1260 DATA 6,102,0,6,0,6,6,102
RA 1270 DATA 6,0,6,0,6,102,6,102
VQ 1280 DATA 0,6,0,6,0,96,0,6
YD 1290 DATA 0,6,6,102,6,6,6,6
PC 1300 DATA 6,6,6,102,0,6,0,6
LT 1310 DATA 0,6,6,102,6,0,6,0
QF 1320 DATA 6,102,0,6,0,6,6,102
MT 1330 DATA 6,0,6,0,6,0,6,102
XR 1340 DATA 6,6,6,6,6,102,6,102
UM 1350 DATA 6,6,0,6,0,6,0,6
TQ 1360 DATA 0,6,0,6,6,102,6,6
GV 1370 DATA 6,6,0,96,6,6,6,6
BF 1380 DATA 6,102,6,102,6,6,6,6
MZ 1390 DATA 6,102,0,6,0,6,0,6
DG 1400 DATA 0,80,5,5,5,5,5,85
FP 1410 DATA 5,5,5,5,5,5,5,85
NF 1420 DATA 5,0,5,0,5,0,5,0
AS 1430 DATA 5,0,5,85,5,85,5,0
OL 1440 DATA 5,0,5,80,5,0,5,0
EM 1450 DATA 5,85,5,85,0,80,0,80
DQ 1460 DATA 0,80,0,80,0,80,5,85
AQ 1470 DATA 80,5,85,5,85,5,80,85
HI 1480 DATA 80,85,80,85,80,5,5,85
CQ 1490 DATA 5,5,5,5,5,5,5,5
ID 1500 DATA 5,5,5,85,5,85,5,5
ZX 1510 DATA 5,5,5,80,5,5,5,5
UG 1520 DATA 5,5,5,85,5,0,5,0
BO 1530 DATA 5,85,0,5,0,5,5,85
XS 1540 DATA 5,85,0,80,0,80,0,80
OG 1550 DATA 0,80,0,80,0,80,0,5
PH 1560 DATA 0,5,0,5,0,5,0,5
PK 1570 DATA 0,5,0,5,0,5,0,5
UM 1580 DATA 0,5,5,5,85,85,80,85
AP 1590 DATA 0,5,5,5,5,5,5,5
FQ 1600 DATA 5,5,5,5,5,5,5,85
FT 1610 DATA 5,5,5,5,5,5,5,85
SB 1620 DATA 0,80,0,80,0,80,104
```

Listing 2.

```
YB 200 POKE 1536,0:DIM B$(256):POKE 1563,0:GOSUB 5340:COL=0:C1=0:AT=0:PL=0:GAME=0
RY 220 GOSUB 3900:GOSUB 3980
EX 240 GOSUB 4120:GAME=GAME+1:START=START-1:START=ABS(START):POKE 204,12:POKE 205,148
YB 260 BYTE=GAME*14:GOSUB 6120:GOSUB 4660:GOSUB 6300:IF START=1 THEN 320
OA 280 GOTO 4700
TJ 300 IF STRIG(0)=0 THEN 300
YU 320 IF MOVE<3 THEN 2280
UU 340 COL=0:GOSUB 2000:BEST=-1
SP 440 IF CENTRE=1 THEN 720
SG 460 MATCH=1:SPACE=0:FOR W=2 TO 3:FOR T=2 TO 3:IF BORD(W,T)=-1 THEN W1=W:T1=T:SPACE=SPACE+1:GOTO 500
TM 480 GOSUB 2040
TR 500 NEXT T:NEXT W:IF SPACE=0 THEN CENTRE=1:GOTO 720
AX 520 IF MATCH=0 THEN 600
CO 540 FOR L=4 TO 1 STEP -1:IF S(L)=-1 THEN NEXT L:L=1
XI 560 BEST=S(L):W=W1:T=T1:GOTO 2140
EY 600 IF SPACE<>1 THEN 720
PU 620 COL=1:W2=W1:T2=T1:IF W1=T1 THEN 980
NJ 640 GOTO 1260
CN 720 IF CORNER=1 THEN 980
DG 740 MATCH=1:SPACE=0
RM 760 GOSUB 2000:FOR WW=1 TO 4 STEP 3:FOR TT=1 TO 4 STEP 3:IF BORD(WW,TT)=-1 THEN W1=WW:T1=TT:SPACE=SPACE+1:GOTO 820
UY 780 FOR K=1 TO 4:IF BORD(WW,TT)=BORD(5,K) AND S(K)<>-1 THEN S(K)=-1:K=5:NEXT K:GOTO 820
OP 800 NEXT K:MATCH=0
WG 820 NEXT TT:NEXT WW:IF SPACE=0 THEN CORNER=1:GOTO 980
VH 840 IF MATCH=0 AND SPACE<>1 THEN 980
CV 860 FOR L=4 TO 1 STEP -1:IF S(L)=-1 THEN NEXT L:L=1
TK 880 BEST=S(L):W=W1:T=T1:IF MATCH=0 AND SPACE=1 THEN COL=4:W2=W1:T2=T1:GOTO 980
MF 900 GOTO 2140
YX 980 IF D1=1 THEN 1260
FU 1000 SPACE=0:MATCH=1:GOSUB 2000:W=1:FOR T=1 TO 4:IF BORD(W,T)=-1 THEN W1=W:T1=T:SPACE=SPACE+1:GOTO 1040
ZF 1020 GOSUB 2040
RD 1040 W=W+1:NEXT T:IF SPACE=0 THEN D1=1:GOTO 1260
ED 1060 IF COL=1 THEN 1100
KH 1080 IF MATCH=0 AND SPACE<>1 THEN 1260
TF 1100 FOR L=1 TO 4:IF S(L)=-1 THEN NEXT L
IX 1120 BEST=S(L):W=W1:T=T1:IF COL=1 THEN COL=0:W=W2:T=T2
CQ 1140 IF MATCH=0 AND COL=4 THEN 1180
QF 1160 GOTO 2140
TC 1180 GOSUB 2000:W=W2:MATCH=1:FOR T=1 TO 4:IF BORD(W,T)=-1 THEN 1190
LH 1184 GOSUB 2040:IF MATCH=0 THEN 1192
SX 1190 NEXT T:W1=W2:T1=T2:COL=0:GOTO 1100
IT 1192 GOSUB 2000:T=T2:MATCH=1:FOR W=1 TO 4:IF BORD(W,T)=-1 THEN 1196
AS 1194 GOSUB 2040
VC 1196 NEXT W:W1=W2:T1=T2:COL=0:GOTO 1100
TV 1260 IF D2=1 THEN 1540
BU 1280 GOSUB 2000:SPACE=0:MATCH=1:W=4:FOR T=1 TO 4:IF BORD(W,T)=-1 THEN W1=W:T1=T:SPACE=SPACE+1:GOTO 1320
ZF 1300 GOSUB 2040
UX 1320 W=W-1:NEXT T:IF SPACE=0 THEN D2=1:GOTO 1540
MN 1340 IF COL=1 THEN 1380
LP 1360 IF MATCH=0 AND SPACE<>1 THEN 1540
UH 1380 FOR L=1 TO 4:IF S(L)=-1 THEN NEXT L
IX 1400 BEST=S(L):W=W1:T=T1:IF COL=1 THEN COL=0:W=W2:T=T2
CQ 1420 IF MATCH=0 AND COL=4 THEN 1180
QF 1440 GOTO 2140
VF 1540 COL=0
SS 1560 FOR T=1 TO 4:MATCH=1:GOSUB 2000:SPACE=0
KE 1580 FOR W=1 TO 4:IF BORD(W,T)=-1 THEN W1=W:T1=T:SPACE=SPACE+1:GOTO 1620
ZL 1600 GOSUB 2040
YP 1620 NEXT W:IF SPACE=1 AND MATCH=1 THEN T=5:NEXT T:GOTO 1700
BQ 1640 IF COL=-2 AND SPACE=1 THEN 1700
MX 1660 IF SPACE=1 THEN COL=-1
ZK 1680 NEXT T:GOTO 1780
FC 1700 FOR L=1 TO 4:IF S(L)=-1 THEN NEXT L:L=3
AN 1710 IF MATCH=0 THEN L=INT(4*RND(0)+1)
AN 1720 BEST=S(L):W=W1:T=T1:GOTO 2140
DE 1780 FOR W=1 TO 4:MATCH=1:GOSUB 2000
FY 1800 SPACE=0:FOR T=1 TO 4:IF BORD(W,T)=-1 THEN W1=W:T1=T:SPACE=SPACE+1:GOTO 1840
ZV 1820 GOSUB 2040
MR 1840 NEXT T:IF SPACE=1 AND MATCH=1 THEN W=5:NEXT W:GOTO 1900
VV 1860 IF SPACE=1 AND COL<>-1 THEN COL=2
QC 1880 NEXT W:IF COL=-1 THEN COL=-2:GOTO 1560
TS 1900 FOR L=1 TO 4:IF S(L)=-1 THEN NEXT L:L=INT(4*RND(0)+1)
AR 1920 BEST=S(L):W=W1:T=T1:GOTO 2140
JM 1960 REM ****************************
YX 1980 IF TEST=2 THEN 2020
FF 2000 FOR K=1 TO 4:S(K)=BORD(5,K):NEXT K:RETURN
XU 2020 FOR K=1 TO 4:S(K)=BORD(K,0):NEXT K:RETURN
HY 2040 FOR K=1 TO 4:IF BORD(W,T)=BORD(5,K) AND S(K)<>-1 THEN S(K)=-1:K=6:NEXT K:RETURN
TJ 2060 NEXT K:MATCH=0:RETURN
MZ 2140 D=BORD(5,G):IF BEST<>-1 THEN D=BEST
SG 2160 B$(Y+30,Y+54)=BL$:X=69+T*20:Y=-9+W*30:XA=T:YA=W:J=XA*2-1+(YA-1)*8
AQ 2180 B$(Y,Y+83)=PL$:POKE 53249,X:GOSUB 5260:GOSUB 5200
SZ 2200 G=G+1:IF G>4 THEN G=1
RM 2220 GOTO 4700
YU 2280 IF MOVE>1 THEN 2400:REM PLAYER WENT FIRST
TF 2300 W=INT(4*RND(0)+1):T=INT(2*RND(0)+2):IF W=1 OR W=4 THEN T=(T=2)+(4*(T=3))
TU 2320 G=4:BEST=-1:XA=T:YA=W:GOTO 2140
UW 2400 FOR T1=1 TO 4 STEP 3:FOR W1=1 TO 4 STEP 3:IF BORD(W1,T1)<>-1 THEN 2540
KV 2420 NEXT W1:NEXT T1
QF 2440 GOTO 340
```

```
GH 2540 G=4:BEST=-1
YQ 2560 IF (T1=1 AND W1=1) OR (T1=4 AND W
   1=4) THEN W=INT(2*RND(0)+2):T=(3*(W=2)
   )+(2*(W=3)):GOTO 2140
QH 2580 W=INT(2*RND(0)+2):T=(2*(W=2))+(3*
   (W=3)):GOTO 2140
YJ 2600 REM * Test scores
DY 2660 COLOR 6:PLOT 72,20:PLOT 72,26:DRA
   WTO 72,34:PLOT 71,22:DRAWTO 73,22:PLOT
    70,24:DRAWTO 74,24
OO 2680 PLOT 71,34:PLOT 71,36:PLOT 73,34:
   PLOT 73,36
UA 2700 FOR I=1 TO 4:BYTE=BORD(5,I)*36:PO
   KE 204,ASC(N$(7+I*2)):POKE 205,ASC(N$(
   8+I*2)):GOSUB 6100
FK 2720 NEXT I
ZN 2740 FOR TEST=1 TO 2
VF 2760 GOSUB 1980:SC=0:FOR W=2 TO 3:FOR
   T=2 TO 3:W1=BORD(W,T)
GD 2780 GOSUB 3600:IF K=5 THEN W=5:T=5
PR 2800 NEXT T:NEXT W:IF SC=4 THEN SC=20:
   GOSUB 4160:GOSUB 3880
JI 2840 REM * Corners
JR 2880 GOSUB 1980:SC=0:FOR W=1 TO 4 STEP
    3:FOR T=1 TO 4 STEP 3:W1=BORD(W,T)
FJ 2900 GOSUB 3600:IF K=5 THEN W=5:T=5
TW 2920 NEXT T:NEXT W:IF SC=4 THEN SC=15:
   GOSUB 4240:GOSUB 3880
IV 2960 REM * DIAG L TO R
YQ 3000 GOSUB 1980:SC=0:W=1:FOR T=1 TO 4:
   W1=BORD(W,T)
KM 3020 GOSUB 3600:IF K=5 THEN T=5
EN 3040 W=W+1:NEXT T:IF SC=4 THEN SC=10:G
   OSUB 4320:GOSUB 3880
HG 3080 REM * DIAG R TO L
BS 3120 GOSUB 1980:SC=0:W=4:FOR T=1 TO 4:
   W1=BORD(W,T)
KU 3140 GOSUB 3600:IF K=5 THEN T=5
DS 3160 W=W-1:NEXT T:IF SC=4 THEN SC=10:G
   OSUB 4400:GOSUB 3880
WH 3180 REM * ACROSS
CG 3220 FOR W=1 TO 4:SC=0:GOSUB 1980:FOR
   T=1 TO 4:W1=BORD(W,T)
KW 3240 GOSUB 3600:IF K=5 THEN T=5
OP 3260 NEXT T:IF SC=4 THEN SC=5:GOSUB 45
   60:GOSUB 3880
LR 3280 NEXT W
DR 3300 REM * DOWN
IM 3340 FOR T=1 TO 4:GOSUB 1980:SC=0:FOR
   W=1 TO 4:W1=BORD(W,T)
ON 3360 GOSUB 3600:IF K=5 THEN W=5
RS 3380 NEXT W:IF SC=4 THEN SC=5:GOSUB 44
   80:GOSUB 3880
JQ 3400 NEXT T
ED 3420 IF TEST=1 THEN POKE 204,147:POKE
   205,132:GOSUB 6140:GOTO 3460
DK 3440 POKE 204,113:POKE 205,132:GOSUB 6
   140:GOTO 3500
HB 3460 COLOR 6:PLOT 5,20:PLOT 5,26:DRAWT
   O 5,34:PLOT 4,22:DRAWTO 6,22:PLOT 3,24
   :DRAWTO 7,24
MZ 3480 PLOT 4,34:PLOT 4,36:PLOT 6,34:PLO
   T 6,36
TN 3500 NEXT TEST:IF GAME=4 THEN 3720
EN 3520 FOR I=1 TO 24:SOUND 0,I*2,10,6:SO
   UND 1,I*4,10,6:SOUND 0,255,10,6:SOUND
   1,255,10,6:NEXT I
UR 3540 SOUND 0,0,0,0:SOUND 1,0,0,0
NL 3560 IF STRIG(0)=1 THEN 3560
ZA 3580 B$(Y+30,Y+54)=BL$:GOSUB 5280:GOTO
    240
HG 3600 IF TEST=2 THEN 3660
NM 3620 FOR K=1 TO 4:IF W1=BORD(5,K) AND
   S(K)<>-1 THEN SC=SC+1:S(K)=-1:K=6
HT 3640 NEXT K:RETURN
JC 3660 REM
GI 3680 FOR K=1 TO 4:IF W1=BORD(K,0) AND
   S(K)<>-1 THEN SC=SC+1:S(K)=-1:K=6
HJ 3700 NEXT K:RETURN
JA 3720 IF AT>PL THEN 3780
BY 3740 COLOR 1:FOR I=4 TO 8:PLOT I,14:DR
   AWTO I,106:NEXT I:COLOR 3:W=6:GOTO 380
   0
IY 3750 IF AT=PL THEN 3780
VK 3760 GOTO 3860
VP 3780 COLOR 1:FOR I=70 TO 74:PLOT I,14:
   DRAWTO I,106:NEXT I:COLOR 3:W=72
JR 3800 FOR I=102 TO 18 STEP -2:SOUND 0,I
   ,10,6:SOUND 1,10,8,6
XE 3820 PLOT W,I:PLOT W-1,I-1:DRAWTO W+1,
   I-1:SOUND 1,10,2,6:NEXT I
UX 3840 SOUND 0,0,0,0:SOUND 1,0,0,0
EW 3850 IF PEEK(53279)<>6 THEN 3850
NW 3860 RUN
AG 3880 IF TEST=2 THEN PL=PL+SC:GOTO 3980
QP 3900 AT=AT+SC:C1=INT(AT/100):C2=C1:FOR
    I=1 TO 3:BYTE=C1*14
CK 3920 POKE 204,190+(2*I):POKE 205,129:G
   OSUB 6120:IF I=1 THEN C1=INT(AT/10):C1
   =C1-(C2*10):GOTO 3960
SK 3940 C1=INT((AT-C2*100)-C1*10)
HJ 3960 NEXT I:RETURN
NP 3980 C1=INT(PL/100):C2=C1:FOR I=1 TO 3
   :BYTE=C1*14
HQ 4000 POKE 204,170+(2*I):POKE 205,129:G
   OSUB 6120:IF I=1 THEN C1=INT(PL/10):C1
   =C1-(C2*10):GOTO 4040
UZ 4020 C1=INT((PL-C2*100)-C1*10)
GM 4040 NEXT I:RETURN
EI 4060 IF TEST=2 THEN PL=PL+SC:GOTO 4100
SX 4080 AT=AT+SC
AE 4100 RETURN
TD 4120 FOR I=0 TO 5:FOR W=1 TO 4:BORD(I,
   W)=-1:NEXT W:NEXT I
OD 4140 H=1:V=1:X=89:Y=21:XA=1:YA=1:PICK=
   0:J=1:MOVE=1:D1=0:D2=0:CENTRE=0:CORNER
   =0:R=0:G=1:RETURN
AB 4160 POKE 203,128:FOR H=8 TO 1 STEP -1
   :POKE 207,155:SOUND 0,32,12,H
NL 4180 C=USR(ADR("h[illegible]"))
HG 4200 IF PEEK(540)<>0 THEN 4200
XW 4220 SOUND 0,16,12,H:NEXT H:GOTO 4640
VE 4240 POKE 203,128:FOR H=8 TO 1 STEP -1
   :POKE 206,70:POKE 207,156:SOUND 0,40,1
   2,H
FU 4260 C=USR(ADR("h[illegible]"))
RC 4280 IF PEEK(540)<>0 THEN 4280
UU 4300 SOUND 0,20,12,H:NEXT H:GOTO 4640
ZT 4320 POKE 203,128:FOR H=8 TO 1 STEP -1
   :POKE 207,155:SOUND 0,32,12,H
BU 4340 C=USR(ADR("h[illegible]"))
PU 4360 IF PEEK(540)<>0 THEN 4360
YQ 4380 SOUND 0,16,12,H:NEXT H:GOTO 4640
JR 4400 POKE 203,128:FOR H=8 TO 1 STEP -1
   :POKE 206,254:SOUND 0,28,12,H
LH 4420 C=USR(ADR("h[illegible]"))
```

```
OM 4440 IF PEEK(540)<>0 THEN 4440
XI 4460 SOUND 0,14,12,H:NEXT H:GOTO 4640
ID 4480 POKE 203,128:FOR H=8 TO 1 STEP -1:SOUND 0,36,12,H
BV 4500 C=USR(ADR("h[illegible]"))
NE 4520 IF PEEK(540)<>0 THEN 4520
ZM 4540 SOUND 0,18,12,H:NEXT H:GOTO 4640
JV 4560 POKE 203,128:FOR H=8 TO 1 STEP -1:SOUND 0,46,12,H
OZ 4580 C=USR(ADR("h[illegible]"))
LW 4600 IF PEEK(540)<>0 THEN 4600
ZE 4620 SOUND 0,23,12,H:NEXT H
YB 4640 SOUND 0,0,0,0:RETURN
NP 4660 FOR W=1 TO 7 STEP 2:D=INT(5*RND(0)):BORD((W+1)/2,0)=D:POKE 204,ASC(N$(W,W))
QH 4680 POKE 205,ASC(N$(W+1,W+1)):BYTE=D*36:GOSUB 6100:D=INT(5*RND(0)):BORD(5,(W+1)/2)=D:NEXT W:RETURN
YG 4700 ST=STICK(0):IF STRIG(0)=0 THEN 4940
JB 4720 IF ST=15 THEN 4700
NE 4730 POKE 77,0
FR 4740 H=(ST=7)-(ST=11)+(ST=6)-(ST=10)+(ST=5)-(ST=9):V=(ST=13)-(ST=14)+(ST=5)-(ST=6)+(ST=9)-(ST=10)
CZ 4760 X=X+(H*20):Y=Y+(V*30):XA=XA+H:YA=YA+V:J=J+(2*H)+(8*V)
WT 4780 IF X>149 THEN X=149:XA=4:J=J-2*H
TH 4800 IF X<69 THEN X=69:XA=0:J=J-2*H
HQ 4820 IF Y<21 THEN Y=21:YA=1:J=J-8*V
LB 4840 IF Y>111 THEN Y=111:YA=4:J=J-8*V
BR 4860 B$(Y,Y+83)=PL$:POKE 53249,X
SS 4880 GOTO 4700
TF 4940 IF PICK=1 THEN 5060
JC 4960 IF X>69 THEN GOTO 5140
JC 4980 GOSUB 5260:PICK=1:D=BORD(YA,0)
GP 5000 W1=YA*2-1:W2=ASC(N$(W1,W1))+60:T2=ASC(N$(W1+1,W1+1))+1:IF W2>255 THEN W2=W2-256:T2=T2+1
QJ 5020 POKE 204,W2:POKE 205,T2:POKE 208,W2:POKE 209,T2
JP 5040 C=USR(ADR("h[illegible]")):GOTO 5160
PQ 5060 IF X=69 OR BORD(YA,XA)<>-1 THEN GOTO 5140
DR 5080 POKE 204,PEEK(208):POKE 205,PEEK(209):GOSUB 5260:PICK=0
LA 5100 C=USR(ADR("h[illegible]"))
YK 5120 GOSUB 5200:GOTO 300
LU 5140 SOUND 0,250,10,8:FOR W=1 TO 20:NEXT W:SOUND 0,0,0,0:GOTO 4700
KL 5160 IF STRIG(0)=0 THEN 5160
SF 5180 GOTO 4700
VH 5200 BORD(YA,XA)=D:BYTE=D*36:POKE 204,ASC(M$(J,J)):POKE 205,ASC(M$(J+1,J+1)):GOSUB 6100:MOVE=MOVE+1
XV 5220 IF MOVE>16 THEN 2600
AT 5240 RETURN
VX 5260 FOR I=10 TO 0 STEP -2:SOUND 0,100,10,I:SOUND 0,85,10,I:FOR W=1 TO 8:NEXT W:NEXT I:RETURN
QC 5280 FOR W=1 TO 7 STEP 2:POKE 203,18:POKE 204,ASC(N$(W,W)):POKE 205,ASC(N$(W+1,W+1))
SM 5300 C=USR(ADR("h[illegible]"))
ML 5320 NEXT W:RETURN
SU 5340 GRAPHICS 10:POKE 704,2:DIM PL$(86),BL$(24),N$(16),M$(32),WORD$(15),BORD(5,4)
JD 5360 RESTORE 5440:FOR I=1636 TO 1646:READ D:POKE I,D:NEXT I:POKE 512,100:POKE 513,6
ES 5380 N$="[illegible]":M$="[illegible]":WORD$="[illegible]"
DX 5400 DIM S(4):BL$(1)="♦":BL$(24)="♦":BL$(2)=BL$:POKE 54286,192
FC 5420 POKE 705,154:POKE 706,252:POKE 707,50:POKE 709,202:POKE 712,92:POKE 710,120:POKE 711,170
MU 5440 DATA 72,169,166,141,10,212,141,23,208,104,64
OB 5460 DL=PEEK(560)+256*PEEK(561)+4:POKE DL+144,143
GD 5480 SCREEN=PEEK(DL)+256*PEEK(DL+1)+6006:SYMBOL=30000:POKE 203,0
ZJ 5500 IF PEEK(1536)<>0 THEN GOTO 5560
FG 5520 GOSUB 6380
GB 5540 RESTORE 6340:FOR I=0 TO 51:READ D:POKE 1536+I,D:NEXT I
PD 5560 POKE 54286,192:COLOR 4:FOR I=0 TO 10 STEP 10:PLOT 0,I:DRAWTO 78,I:NEXT I
GC 5580 PLOT 0,0:DRAWTO 0,9:FOR I=0 TO 9:PLOT 38,I:DRAWTO 40,I:NEXT I:PLOT 78,0:DRAWTO 78,9
DG 5600 COLOR 2:REM
CV 5620 FOR I=14 TO 134 STEP 30:PLOT 19,I:DRAWTO 59,I:NEXT I
UM 5640 FOR I=19 TO 59 STEP 10:PLOT I,14:DRAWTO I,134:NEXT I
CN 5660 COLOR 5:FOR I=145 TO 177 STEP 8:PLOT 2,I:DRAWTO 18,I:PLOT 22,I:DRAWTO 38,I
MD 5680 PLOT 42,I:DRAWTO 58,I:PLOT 62,I:DRAWTO 78,I:NEXT I
DZ 5700 FOR I=2 TO 18 STEP 4:PLOT I,145:DRAWTO I,177:PLOT I+20,145:DRAWTO I+20,177
BB 5720 PLOT I+40,145:DRAWTO I+40,177:PLOT I+60,145:DRAWTO I+60,177:NEXT I
GV 5740 COLOR 4:W=4:FOR I=148 TO 172 STEP 8:PLOT 8,I:PLOT 8,I+1:PLOT W,164:PLOT W,165:PLOT W+20,I:PLOT W+20,I+1
SA 5760 PLOT 40-W,I:PLOT 40-W,I+1:W=W+4:NEXT I
LD 5780 FOR I=148 TO 172 STEP 24:PLOT 44,I:PLOT 44,I+1:PLOT 56,I:PLOT 56,I+1:NEXT I
BE 5800 FOR I=156 TO 164 STEP 8:PLOT 68,I:PLOT 68,I+1:PLOT 72,I:PLOT 72,I+1:NEXT I
KZ 5820 COLOR 3:PLOT 0,141:DRAWTO 79,141:DRAWTO 79,188:DRAWTO 0,188:DRAWTO 0,141
GN 5840 FOR I=2 TO 1 STEP -1:COLOR I:PLOT 72-I,112
DL 5860 DRAWTO 72-I,134:DRAWTO 76+I,134:DRAWTO 76+I,112:DRAWTO 72-I,112:NEXT I
AG 5880 POKE 204,20:POKE 205,158:BYTE=70:GOSUB 6120
CI 5900 POKE 204,29:POKE 205,158:BYTE=14:GOSUB 6120
AW 5920 POKE 204,31:POKE 205,158:BYTE=0:GOSUB 6120
```

```
DJ 5940 POKE 204,39:POKE 205,158:BYTE=14:
   GOSUB 6120
BW 5960 POKE 204,41:POKE 205,158:BYTE=70:
   GOSUB 6120
LM 5980 POKE 204,49:POKE 205,158:BYTE=28:
   GOSUB 6120
BD 6000 POKE 204,51:POKE 205,158:BYTE=0:G
   OSUB 6120
SW 6020 FOR I=1 TO 5:BYTE=ASC(WORD$(I,I))
   *14
EH 6040 POKE 204,159+(2*I):POKE 205,129:G
   OSUB 6120:NEXT I
FN 6060 FOR I=6 TO 10:BYTE=ASC(WORD$(I,I)
   )*14
JF 6080 POKE 204,169+(2*I):POKE 205,129:G
   OSUB 6120:NEXT I:GOTO 6160
ZZ 6100 C=USR(ADR("hhh[illegible]u[illegible]u[illegible]
   i&[illegible]"),BYTE):RETURN
NH 6120 C=USR(ADR("h[illegible]h[illegible]u[illegible]he[illegible]
   [illegible]&[illegible]"),BYTE)
   :RETURN
OK 6140 C=USR(ADR("h[illegible]
   i([illegible]h[illegible]h[illegible]")):RETURN
IW 6160 I=PEEK(106)-40:POKE 54279,I:POKE
   53257,1:POKE 53277,3:W=I
EF 6180 VTAB=PEEK(134)+PEEK(135)*256:ATAB
   =PEEK(140)+PEEK(141)*256:OFF=I*256+128
   0-ATAB
OQ 6200 HI=INT(OFF/256):LO=OFF-HI*256:POK
   E VTAB+2,LO:POKE VTAB+3,HI:POKE 559,62
   :POKE 623,163
TI 6220 FOR I=1 TO 200 STEP 10:B$(I)=BL$:
   NEXT I
DK 6240 B$(1)="♦":B$(256)="♦":B$(2)=B$
MV 6260 PL$="♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
   ♦♦▸A♦A♦A♦A♦A♦A♦A♦A♦A♦A▸♦♦♦♦♦♦♦♦♦♦♦♦♦♦
   ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦"
WR 6280 START=INT(2*RND(0)):GOSUB 4140
UN 6300 B$(Y,Y+83)=PL$:POKE 53249,X:RETUR
   N
IG 6320 REM *M/C CODE
OT 6340 DATA 104,169,228,133,206,104,170,
   105,117,133,207,24,104,101,206,133,206
   ,144,5,138,101,207,133,207,160,0
CD 6360 DATA 177,206,145,204,200,177,206,
   145,204,200,24,165,204,105,38,133,204,
   144,2,230,205,192,14,208,231,96,0
AM 6380 BYTE=503
PB 6400 OPEN #1,4,0,"D:FONT"
JH 6420 C=848:POKE C+2,7
GL 6440 H=117:L=48:POKE C+4,L:POKE C+5,H
GX 6460 H=1:L=247:POKE C+8,L:POKE C+9,H
RX 6480 D=USR(ADR("hhh[illegible]_V[illegible]"),16)
DQ 6500 CLOSE #1:RETURN
```

PUBLIC DOMAIN SOFTWARE
AT AFFORDABLE PRICES
ONLY £5 PER PACK!

Pack A	Pack B	Pack C
Packed full of desk accessories and utilities.	Word processing programs and utilities.	Lots of communications packages, inc KERMIT.
Pack D	**Pack E**	**Pack F**
Games, Games, and yet more Games!	Adventure games, including Hack and Colossal Cave.	Graphics utilities, image processor, display programs.

Each pack consists of 3 x 3.5" disks with over 300K of data on each disk!

10 x 3.5" Single-Sided Disks
Unbranded - Fully Guaranteed
ONLY £9.50

PLEASE SEND CHEQUE OR POSTAL ORDERS TO...
Highfield Supplies
Dept STW1
16 Nursery Road
Bradford
BD7 4LH

Atari ST

Please send S.A.E. for details of our full product range and information pack.

ST STOCKSOFT ST

520 STFM, for latest new lowest prices, send for list

Public Domain software, games, discs and utilities, £5 per disc, post free

HAPPY ENHANCEMENT NOW AVAILABLE FOR THE ST

For Free Catalogue for the above plus more send an S.A.E. to:

STOCKSOFT *(DEPT ST)*
15 Woodbrooke Road, Birmingham B30 1UE

XL/XE MOUSE

AN ST MOUSE FOR YOUR XL/XE

Includes: Mouse, Art Package, Fast picture loader, Atari Artist/GAD translator. 2 demo programs show the use of the mouse in your own software. Supplied with all operating information.

All for £29.95 Post free

Please state 800XL 130XE and cassette/disk

LETTER 80

©1988 DISCO ™

80-COLUMN WP ON SCREEN

This full-feature letter/word processor supports standard Atari keyboard editing PLUS wordwrap, insert & justification. Address, date and ref. area also control search/mailshot feature. Mouse/joystick operation with pull-down menus.

Disk only £24.95 Post free

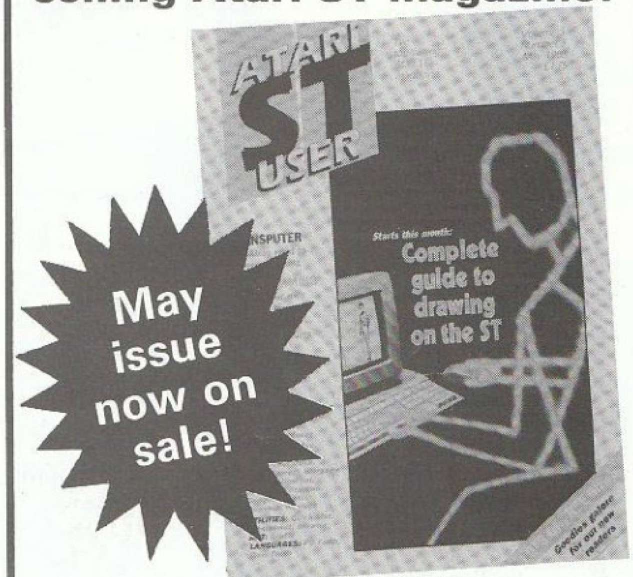
Buy both for only £39.95 post free

COD service available for £1.50

GLOBAL COMPUTER SERVICES

108 Navigation Road, NORTHWICH, Cheshire CW8 1BE (0606) 782413

Don't miss out! Keep right up to date with all the latest developments in the exciting world of the Atari ST – with Britain's top-selling Atari ST magazine!



Among the many features you'll find in the May issue:

- Languages: Gem Forth and PD Forth examined.
- Business: MasterPlan, Partner, Timeworks DTP, Certificate Maker reviewed.
- Programs: Create custom-designed mouse cursors. How to model tensile structures.
- Games: Xenon, Oids, Eye, Mission Elevator, Tetris, Test Drive, Crash Garrett, Wargame Construction Set, Terramex, Slaygon, Dungeon Master, Gunship and Warlock – all reviewed.
- Features: Music on the ST. Top art in Gallery. Transputer technology revealed. Pete Johnson, Rob Hubbard and CRL interviews.

Available NOW from your newsagent or larger computer dealers for £1.75.

Alternatively, take out a subscription and get all this *FREE:* An Atari calculator, the Atari ST User Compendium disc, and two discs crammed with public domain software – a bundle worth over £17!

Please send me the next 12 issues of *Atari ST User* for £19 starting with the ______________ issue.

I enclose a cheque made payable to Database Publications.

☐ Please debit my Access/Visa card no:

□□□□ □□□□ □□□□ □□□□

Signed: ______________ Expiry date: ______________

Name ______________

Address ______________

______________ Postcode ______________

Send to: Atari ST User, Europa House, Adlington Park, Adlington, Macclesfield, Cheshire SK10 4NP.

Phone orders: 0625 879920
Telecom Gold: 72:MAG001
Prestel: Key *89 then 614568383

Don't forget to include your credit card number and full address. A420

I 1557/36I


abc
Visit one of the U.K.'s LEADING ATARI DEALERS
Our range of Atari products is extensive, whatever you need Ring, We Can Normally Help
DISKS
10 Disks + Library Box £13.95
30 Disks + 40 cap. Lockable Disk Box £39.95
30 Disks + 80 cap. Lockable Disk Box £43.95
3.5" DS/DD DISKS
Why not have a day out and visit us, all 8 and 16 Bit users made welcome, the Coffee is always Hot!
EPSON LX 800 £275.00
STAR LC 10 £229.00
STAR LC 10 COLOUR £279.00
START Special MUSIC & GRAPHICS Edition
START The ST Quarterly Magazine with FREE DISK £12.95
START
REGULAR ORDERS TAKEN
BACK ISSUES AVAILABLE
All printers come complete with Printer Stand, Lead and Paper, ready to use on your ST.
CITIZEN 120D £199.00
CITIZEN LSP100 £249.00
2 Year Warranty inc.
Hawk Scanner £1,250.00
SAM Digitiser £249.00
NEC P2200 £395.00
Demonstrations of all Atari Business and Desk Top Publishing systems - Scanning and Laser Printing services available
WIZARD ACCOUNTS SOFTWARE
NOW IN STOCK
£249.00 + VAT
VISA
(0533) 700190
(24hr Answerphone)
BBS (0533) 700914
Norfolk (0328) 710316
All prices include VAT and postage in the UK
Cheques & postal Orders made payable to: abc
abc 135 St. Leonards Rd, Leicester, LE2 3BZ

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.

ST LIBRARIAN'S REPORT

Once again I would like to thank all you good people out there who have been supporting the Library. First of all there are those members who have access to PD software, often from many new sources, who have very kindly sent me this material for inclusion into OUR Library. I emphasised OUR because it is there for your benefit, not for mine! The contributions help to swell the Club's funds which finance events such as Exhibition attendances. For the average show, the cost to US the members, through subs etc., is in excess of £700! Secondly, the user. I try very hard to provide disks which you have requested plus those that I think are suitable, both in quality and in content. In fact, there are two programs in the latest additions that were requested - firstly a MONOCHROME emulator. For those members out there who only have colour systems, there are two programs which will allow MONO only programs to be run on colour monitors/TV. There are some speed restrictions, but the end result is quite acceptable! I have tried them out on a number of MONO progs, such as the MRUS series and they all work extremely well!

The second program is a PD SPREADSHEET! This program has quite a number of very useful features found in expensive commercial programs. The MRUS series of four disk are directed to MIDI enthusiasts, in particular owners of the DX7 series of synthesisers/expanders from Yamaha. Most of these use an Artificial Intelligence shell developed by the author, Martin Russ. These progs include an Editor, with restricted functions from the commercial prog, a Random Voice Generator, a MIDI channel analyser, a System Exclusive Analyser, etc.

There are also some very good games in this release. There is a very difficult shoot 'em up called DAMONOID. There are some ACCESSORY disks, always popular. Some source files for BASIC enthusiasts - GFA and ST - can also be found.

In most instances, these disks have been sent in by members. Can you contribute??? Remember that you can pick any disk from the Library in exchange for a disk contributed! No fees, nothing!

The Library is expanding at such a pace now, I have split it into two files. Remember, you will need to read them with the help of the TEMPUS demo that is included on the library disk. The reason for this change is to enable the 1/2 MEG system owner to read all the files without running out of memory!

Here are the latest additions.

ADEMO 3

A mammoth, 4 hour plus graphics/sound demo - great! 1/2 MB auto-boot colour.

ADEMO 6

Includes: Skymap, Murray, Kal. 1/2 MB mono only.

ADEMO 7

Includes: Kiss, Car Show. 1/2 MB colour.

ANIM 5

Very good animation demo of a running tap! 1/2 MB colour.

ART 16

22 items, 345K Tiny Pics - No Tiny View program to optimise disk space. Includes: Nassa, Pyram1, Mark1, etc. 1/2 MB colour.

ART 17

26 items, 342K Tiny Pics with Tiny View. Includes: Tinafin, Whiplash, Supman, etc. 1/2 MB colour.

COMDEMO 3

PFS, commercial demo of meteoroligical satellite program with Degas pics, MOST INTERESTING!!! 1 MB colour.

DISKACC 2

18 items, 254K includes: Hexrech, Breakout, Select. 1/2 MB.

DISKACC 3

43 items, 324K includes: 1-Filepr, Fortune. 1/2 MB.

GAMES 4

27 items, 340K includes Daleks, Dragon, Warzone. 1/2 MB colour.

GAMES 3

14 ITEMS, 327K, Damonoids. 1/2 MB colour.

LGFA 2

26 items, 330K includes Solaterm, Gemvdi. 1/2 MB.

LSTB 2

80 items, 310K includes Games, Demos and Utilities. 1/2 MB.

LSTB 3

89 items, 290K includes MIDI, Music, Tutorial, Games, Demos, Utilities. 1/2 MB.

MMS 3

Pics and Music Studio files similar to SAS series. Includes Brandbg, Mona, Tiger. 1/2 MB colour.

MRUS 1

DX7 voice randomiser. VERY GOOD! 1/2 MB mono only.

MRUS 2

Channel Scope. VERY USEFUL! 1/2 MB mono only.

MRUS 3

MIDI Mon - System exclusive monitor. 1/2 MB mono only.

MRUS 4

AI voice editor for DX7 (limited version) - BRILLIANT!! 1/2 MB mono only.

MIXT 2

25 items, 295K includes: Accessories, Utilities, Tiny Pics. 1/2 MB.

MIXT 3

Games - Invaders, Juggler, Megagame, Mono Emulator, Pics. 1/2 MB.

SOUND 12

Sound Sample that lasts for ever! Hit Reset to stop! WHAM! 1/2 MB colour.

UTILS 10

25 items, 319K includes Barrel, Neo, Palart (BRILLIANT), RDC, Seced and the SPREADSHEET!!! 1/2 MB.

Requests should be sent to Mike Stringer, P.O. Box 3, Rayleigh, Essex SS6 8LR. Make cheques/postal orders payable to 'The U.K. Atari Computer Owners Club'.

ST LIBRARY

READ ALL ABOUT IT!

Atari ST Internals 2nd Edition

Authors: Gerits, Englisch & Bruckman
Publishers: Abacus Software
Cost: £14.95
Review by Bookworm

This 470 page book goes a long way into describing, in a generalised manner, the functions of all the important components that are found in the ST. There are over thirty illustrations to clarify the text. The first comment I would like to make is that it is very readable, considering the authors are German. The translators have done an excellent job.

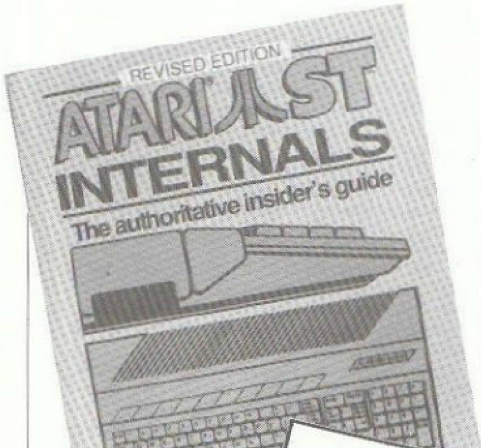
The book is divided into three distinct parts: the integrated circuits, the interfaces and the ST operating system. Additionally, there is an extensive, commented, read-out of the BIOS!

No fewer than thirteen pages are devoted to the 68000 processor - quite a feat. This IC is an amazing chip, but all the important features are covered in a very brief, comprehensive manner. I did not notice any that were omitted.

In a similar, efficient manner all the ICs are covered. Registers, where applicable, pin-outs, flags and similar programmer's requirements are carefully explained. A lot of this information is readily available from data sheets, but the specialised, custom chips are also covered. These are, of course, the GLUE, MMU, DMA and SHIFTER. A great deal of emphasis has been placed on how their multitudenous functions overlap and how they are united.

I found the third chapter which covers the ST Operating System to be very interesting. It covers GEMDOS, Memory, files, BIOS functions, Graphics, Line-A variables and opcodes, the line-F emulator, the VT52 emulator, the 68000 instruction set, etc.

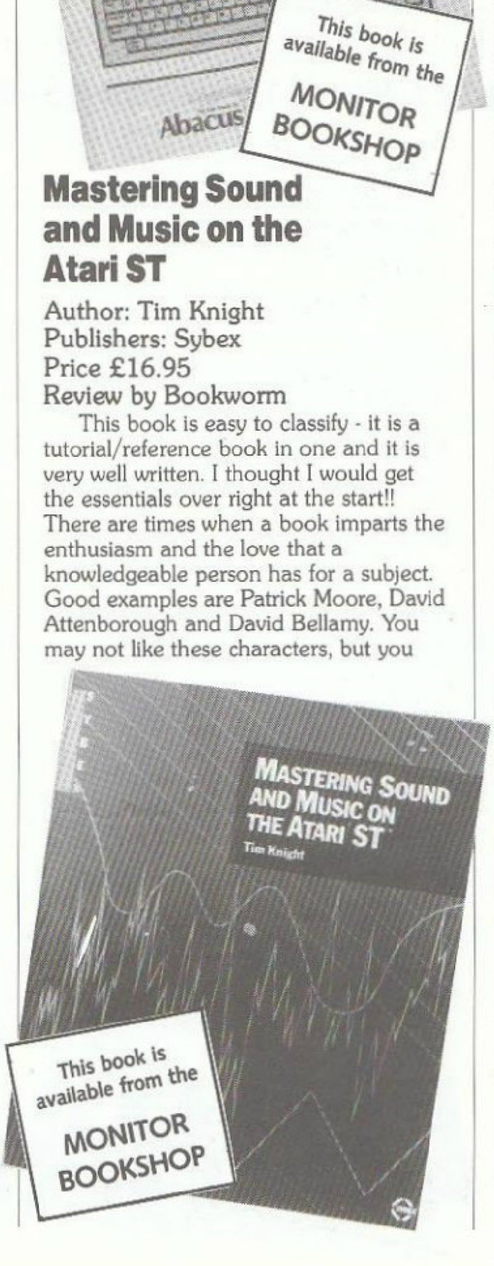
The final, interesting feature is the BIOS listing. As I mentioned earlier, it is commented. This is a most commendable achievement. All in all, this is quite a useful book to have on the shelf. It is a valuable guide to the student, programmer, novice and expert alike. I would have liked the page header indicate the current page topic. It is very hard to classify - it isn't really a text book or a reference book, it is more of a vade mecum.



Mastering Sound and Music on the Atari ST

Author: Tim Knight
Publishers: Sybex
Price £16.95
Review by Bookworm

This book is easy to classify - it is a tutorial/reference book in one and it is very well written. I thought I would get the essentials over right at the start!! There are times when a book imparts the enthusiasm and the love that a knowledgeable person has for a subject. Good examples are Patrick Moore, David Attenborough and David Bellamy. You may not like these characters, but you



cannot escape the magnetic charisma they exude. Tim Knight strikes me as such a person.

Music has different meanings for different people. Some can take it or leave it, some detest it, most enjoy it. The strange thing, most computer enthusiasts belong to the first group, this book is definitely for the latter!

This 270 page book is packed with theory, good programming examples, software/hardware appraisal and much more. The only P.I.T.B. is the programming language is centred around ST BASIC, regular readers know my feelings on this subject. I am getting quite good at doing translations across to FAST and GFA!!

There are eight chapters. Acoustics, Music and the Atari ST; Making Music; Sound Programming; Polyphonic Music; Sound and Music Effects; Your own music maker; MIDI Applications on the Atari ST; Synthesisers and Software.

There are also three Appendices covering such topics as Books, Magazines, Glossary, etc.

The first few chapters cover the theory, including the physics, an introduction to the WAVE and SOUND commands and an introduction to programming. The next few chapters get down to programming, over forty listings are included, all of which are well described and commented. Included in these programs is a large program which allows you to enter and playback your own music. The chapter devoted to MIDI and the following one on software and synthesisers expands the sound capabilities to external devices, a definite improvement to the sound capabilities of the ST!!

It is interesting to read the initial comments of the author regarding the speed of BASIC when attempting to harness the ST's power potential over MIDI. What a shame he did not include some programming examples using C or Modula II, languages that are considerably faster than ST BASIC. Perhaps he will write a book on this subject using programming examples covering these, and possibly Assembler, as well??

Sybex have attained quite a good name for quality products, this book is no exception. Of all the computer orientated books I have read, or appraised, this gets my vote for 'Book of the year'.

GFA Basic - Advanced Programming

Author: Frank Ostrowski
Publisher: Glentop
Price: £15.95
Price with diskette: £19.50
Reviewer: Bookworm

This book is intended for the serious programmer, but it should not be ignored by the beginner. The author of this book is also the author of GFA Basic. Who better to introduce you to the power of one of the ST's premier Basics?

Here is a technical manual, reference book and tutorial wrapped into one, 200 page plus book. Condensed into its seven chapters are topics such as: GRAPHICS, GEMDOS, XBIOS, BIOS, AES LIBRARY FUNCTIONS, RESOURCE (.RSC) FILES, WINDOWS.

The author initially takes you through OPTIMISATION. The logic behind this is twofold. Firstly, your program will run faster and secondly, it will be a better, more professional and efficient program. You will be a more proficient programmer as a result. This is done by skilful appraisal of topics such as TITLE SCREEN, DISK DATA, SORTING, MINIDATA and CALCULATIONS problems, which if not handled correctly, can lead to BIG problems!

The GRAPHICS chapter covers the four GRAPHMODE commands, saving graphics to disk, BLITTING, etc.

The third chapter is a little bible all to itself. TIPS AND PROGRAMMING. More than sixty page are crammed full of hints and tips which include SOUND, DIALOGUE BOXES, FILE COPYING, DISK DIRECTORY, etc.

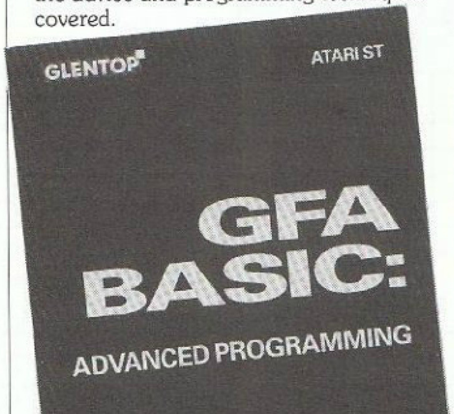
The following seventy pages explain in great depth the commands associated with GEM, BIOS, XBIOS and the associated AES and VDI.

Next RESOURCE FILES are explained in depth and clarity. All you need to know about the creation and useage of menus - bars, boxes, etc. A thorough explanation of Resource Construction Sets covers menus, alert boxes and object trees.

The final chapter, WINDOW leads into a demo program that demonstrates all the many characteristics of WINDOWS: moving, re-sizing, gadgets, slide bars, text and graphic pictures.

Throughout the book there are hundreds of programming examples, mostly small - but some of them are quite large. You will notice that at the heading to this review I mention that an optional diskette is available from the publishers. This contains eighty four of the most useful and I would recommend the purchase of the disk to save you the toil of hours of typing. I noticed that there were a few typographical errors in the book's listings.

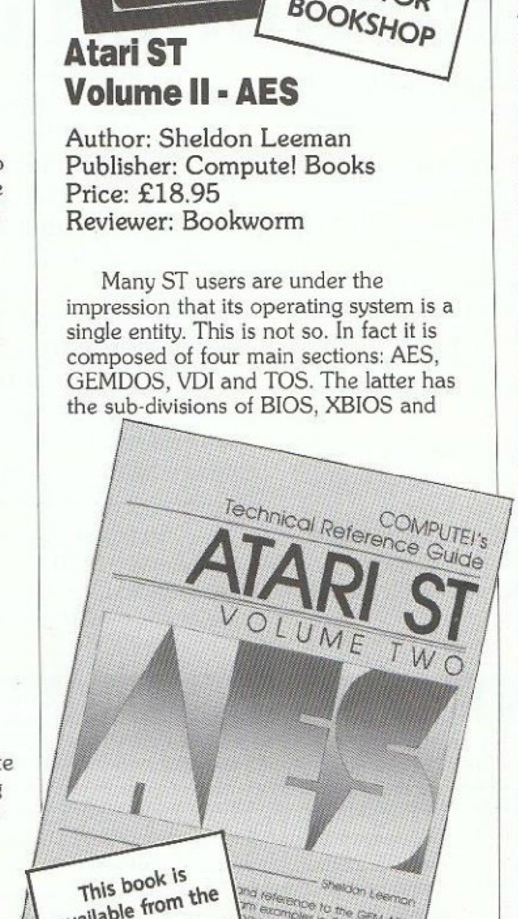
This book is very well written, I was hoping I could congratulate the person responsible for the translation from the original German into English - but I could not. The translation is excellent - it is very readable and what greater authority could one have than Mr. Ostrowski? VERDICT: A most valuable book for the shelf of all GFA Basic programmers, beginner and expert alike. I know that you will greatly benefit from the advice and programming techniques covered.



Atari ST Volume II - AES

Author: Sheldon Leeman
Publisher: Compute! Books
Price: £18.95
Reviewer: Bookworm

Many ST users are under the impression that its operating system is a single entity. This is not so. In fact it is composed of four main sections: AES, GEMDOS, VDI and TOS. The latter has the sub-divisions of BIOS, XBIOS and the LINE A routines.



Volume I of this series of three tutorial/reference books is concerned with the VDI - Virtual Device Interface, the third is probably GEMDOS, I am not sure because I did not receive their listing.

This book is devoted to the AES - Application Environment Services. The AES is part of GEM and among the features that are covered in this extensive and exhaustive book are the following:

APPLICATION and windows
EVENTS
GRAPHICS
RESOURCE FILES and MENUS
OBJECT HANDLING
DESK ACCESSORIES

The author has chosen to illustrate the programming examples with three languages: C, Machine Code and Basic. The 'C' compiler is ALCYN, although MEGAMAX could also be used. For LATTICE users, you will appreciate that the former compilers use 16 bit integers. The 32 bit integer of LATTICE means that SHORT should be used whenever INT appears and everything should be honky dory.

The heart of the AES is the system of libraries. There are eleven such libraries: APPLICATION, EVENT, MENU, OBJECT, FORM, GRAPHICS, SCRAP, FILE SELEctor, WINDOW, RESOURCE and SHELL. Perhaps the easiest way to appreciate the concept of the library is to consider them collections of routines that are called into your program. All such routines are to be found in this book. There is also one of the appendices devoted to a reference and cross reference of all functions by name and opcode.

Another contains the necessary RESOURCE FILE information used to support four of the program listings in the book. It is a great pity that there is not a disk of the program listings available as an optional, as is to be found with the GFA Basic tutorial. Some of the listings are very large and when you consider that three languages are covered, it would have been invaluable.

The author has produced one of the best tutorials I have had the pleasure to read. The programming examples have been chosen to demonstrate the particular functions with clarity and I did not find any typographical errors in the 'C' listings that I examined.

I wish that I had the opportunity to examine the other books in this trilogy, thinking it to be unfair to take one book in isolation, especially this particular one. If the other two are written to this same standard, I will not hesitate in getting them to add to my own bookshelf. I will try to get hold of the other two, hopefully for inclusion in the next edition of MONITOR.

VERDICT: An invaluable wealth of information and programming examples.

CHESSBASE Corner

Correspondent: Michael Stringer

Welcome to what I hope is going to be a regular feature in MONITOR, dedicated primarily to owners of ChessBase. I hope that the news, views and reports, etc. will also be of interest to all chess enthusiasts. To start off, here is a brief resume of the current chess scene, starting off with two very interesting Exhibition Matches.

On February 14th, St Valentine's Day, the Cannes Massacre took place. Picture, if you would, Gary Kasparov seated at a terminal in this very beautiful French city on the Cote d'Azure. His opponents are ten Junior International Champions, seated at similar terminals situated around the world: U.S.A., Canada, Australia, Japan, Switzerland, Italy, Belgium, Soviet Union and London.

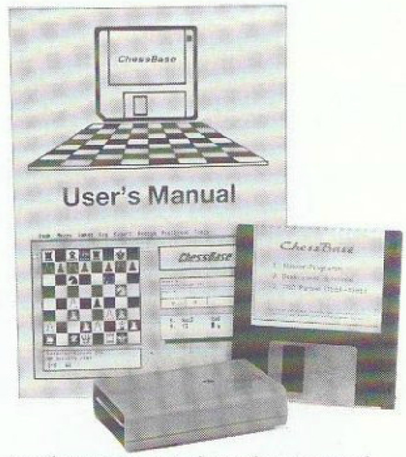
When one pauses and appraises the strength of the opposition, it is comparable to Diago Maradona taking on Liverpool, single handed! All the matches were against the clock and Gary succeeded in winning with the score 8 1/2 to 1 1/2.

One match drawn, one lost. Pretty spectacular. The loss was to a certain Michael Adams, OUR Junior Champion! Although our National Sports people are going through a pretty rough patch at the moment, and that is putting it mildly, British strength in chess is considerable.

A couple of weeks later, in the second Exhibition Match, Mr. Kasparov took on the Manhatten Chess Club. Or rather, their top six players. Like the previous match, this was also simultaneous, against the clock. The opposition proved to be much stronger, but Kasparov survived a powerful attack, coming through to win 4 - 2! Incidentally, one of his opponents was Patrick Wilff - ELo rating 2530! Patrick is capable of some flashes of brilliance and is currently one of the strongest players Stateside.

ChessBase was used by Kasparov during his preparation for these matches. He examines between one hundred and one hundred and twenty games per hour, including note-taking. One sheet of paper is used for each opponent. Considering that the average game is about forty moves, that represents a full half second per move (one per player) AND note taking, where necessary.

On a more serious note, the most prestigious tournament has just concluded. I refer, of course, to the Euwe Memorial Tournament held in Amsterdam. The winner was none other than our very own Nigel Short. On his way to victory, two very notable scalps were KARPOV and Jan TIMMAN! Especially when one considers that these



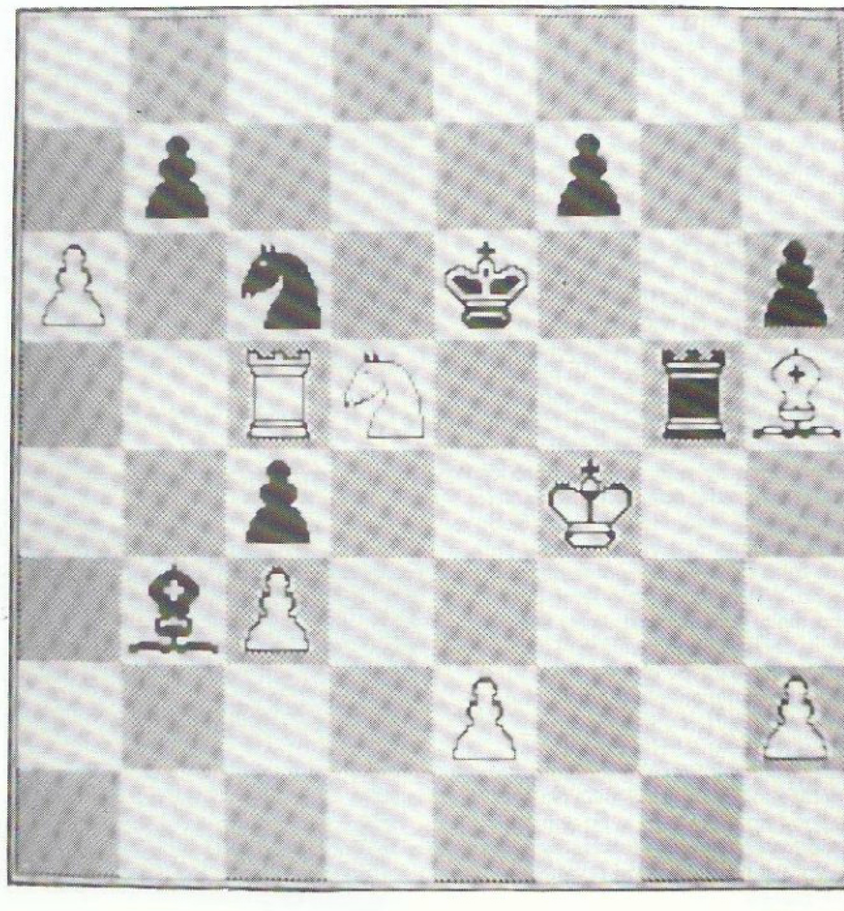
gentleman are rated number two and three in the World Ratings! In a couple of months time, when the next World Ratings are published, Nigel's new rating should be 5! I know that Nigel uses ChessBase, but whether it was used in his preparation for this tournament, I do not know. I would hazard a guess that he did. I think that our congratulations should go out to Gary, Michael and Nigel.

It is quite possible that Nigel may qualify for the next World Championship Finals. That would indeed attract a great deal of interest in chess in this country. Kasparov versus Short. It has a very nice ring to it. This is not wishful thinking, Nigel stands a very good chance of qualifying. One of his fellow competitors is Britain's Jon Speelman! I did say that there is an enormous strength in British Chess! Well, that's the news. For views, comments, advice, help, forthcoming ChessBase products, etc., that will depend upon others. That includes you! To help me, in London there is Louise McDonald and in Germany, Frederic Friedel. I cannot do it single-handed, I am going to need a lot of help!

I will start it off with a plea to Frederic - how about the matches I have just mentioned appearing in a forthcoming ChessBase Magazine????

To conclude this little epistle, the adjacent photograph is the situation from a game in the latest Chessbase Magazine I have, number four. I am not saying which match it is, that would make it too easy for ChessBase owners! They will have to check all 1,000 plus games to try and find it!!!

White is to move and mate in three. A small prize will be given to the winning entry, if any. Please send it to me at the address in the front of the magazine. See you in the next edition.



Predator

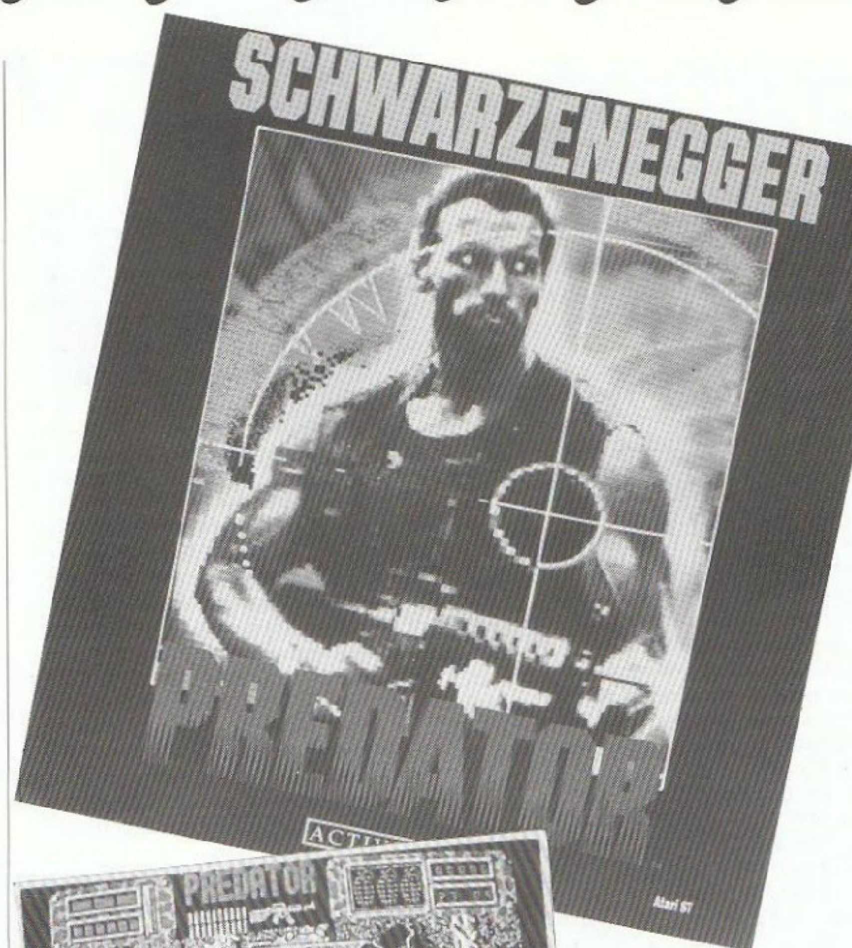
From Activision
Price £14.95
Review by Stuart Rennie

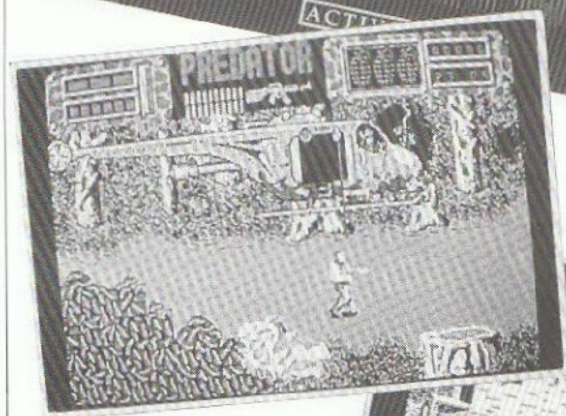
If you've seen 'Arnolds' latest blockbusting movie you'll know exactly what to expect from 'Predator - the Computer Game'. If not, then here's the story so far: Major Alan Schaefer (Arny) is sent on a rescue mission in the jungle looking for surviving crashed helicopter passengers and some Presidential aides. He takes up rear guard position for his six buddies who forge on ahead. Stealth is the name of the game, cautiously you move Arny forward, but things are not going to plan, you start to find the mutilated bodies of your comrades, disfigured in a way that not even enemy guerillas would be capable of! Suddenly you are ambushed by said guerillas, but in your usual skilled, battle hardened way, you despatch these inferior forces to computer heaven (or hell). As you get deeper into the jungle however, you begin to realise that something else is out there, something more deadly than any human force you could ever face! And it's after you! You thought you were the hunter, the one that was giving chase, the predator! You're wrong! It's the PREDATOR and it's coming...

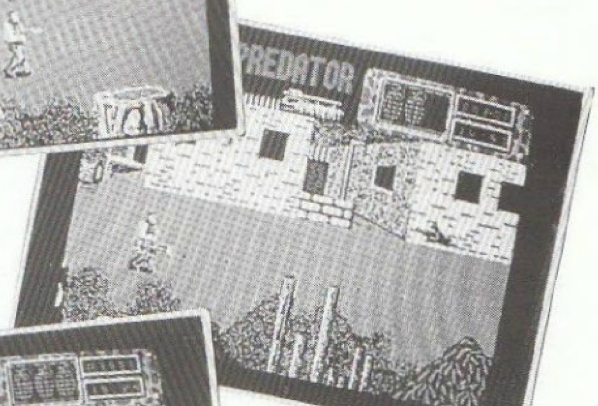
You control Arny with a joystick and some keyboard presses. All the usual movements are there including jump and crouch, also moving the joystick with the fire button held down fires his weapon in the various directions (if he has no weapons then kick, block and punch are available). Arny has some grenades too and he can throw these by pressing left hand Shift for a short throw, or the Space Bar for a long throw, also to pick up or put down an item you can use Return. Each weapon has a limited number of bullets, but you can pick up a new weapon from your fallen comrades.

At the start you are dropped off by helicopter (rotor blade sound effect very good by the way) and progress from left to right across the screen. The graphics are top quality and a joy to watch. Soon hidden enemies pop up to bar your path, some from behind trees, some out of the ground. Watch your ammo level, if the opportunity comes to get a new weapon, take it. Each weapon is different and gives a different type of fire power, the sound the gun makes is changed also (nice touch that). If you hear a rustle in the bushes, beware! The Predator is closing in, the screen changes colour and a triangle forms (which is the creatures weapon starting to zero in on you) and you must run for your life!

Activision have a good one here. More please!







Dungeon Master

From Mirrorsoft
Price £24.95
Review by Colin Thoms

After playing this game for a few weeks I can honestly say it's the best graphic adventure (or should I call it a fantasy role-playing game) I have yet come across on the ST. Although marketed by Mirrorsoft, it comes from FTL in San Diego who gave us Sundog some time back. Congrats where it's due.

On loading the game, you are shown the entrance to the dungeons wherein lies the Firestaff, the object of your quest. You enter into the Hall of Champions where the souls of warriors who perished in the dungeon are imprisoned in mirrors all around the walls. You can choose any 4 from a selection of 24 characters. By pointing to the mirror and pressing the mouse button each champions qualities are shown, these include health, stamina and mana (magical energy). Each champion may already have some possessions, and these are shown graphically (in fact everything in this game is shown graphically, that's the beauty of it), there are also many empty spaces into which things can be put later, as more possessions are obtained. The body of the hero is depicted and these too can have possessions, so you can practically change the whole attire of your characters. An eye is also shown and if you pass objects over it they will be examined in detail, and if you pass something like an apple over the mouth it chomps down and actually eats it, great!

Having selected your 4 (choosing four different types of character is probably best, i.e. a warrior, a ninja, a priest and a magician) you can move on into the depths of the dungeon. You will see a realistic 3-D image of the corridors and rooms, and as you move along you observe things like lanterns, hooks, slime on the floor, handcuffs, etc. You should be looking for things like keys (to open doors), trapdoors, food, more weapons, and anything useful, and also keep listening! How your champions fair is up to you. You should be trying to build up your party's characteristics while you discover how to deal with mumified monsters and figure out the puzzles. You need to 'fluxcage' and 'fuse' the Chaos Lord in order to finish the game.

There are more than 13 levels, most fitting within a 32 x 32 grid. Making a map can be useful to you, but there are areas that cannot be mapped in the 'traditional' sense. A useful way of checking ahead is to use 'magic' vision. Get your characters to practice their skills when there is no danger about, they will improve with practice. Practice at spell making is also a good idea.

Dungeon Master is addictive, stunningly graphical, an object lesson to all other software companies, this is how it should be done! D & D freaks will flip over this, people who previously were not interested in fantasy will come to love it with this one! It has got to be THE game of 1988 so far.

Dizzy Wizard

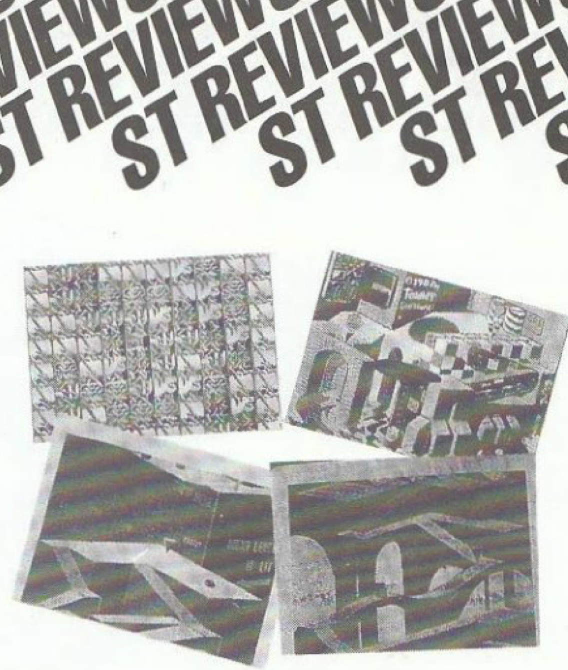
From Robtek
Price £19.95 (needs 1 Meg)
Review by Marvey Mills

Ah... Dizzy Wizard. My, how pleased I was to be asked to review the latest offering from Diamond Games. You see, at the last Atari Show, between mouthfuls of warm lager and stale cheese rolls, I spied this game being demonstrated on the Robtek stand (Robtek are the distributors for Diamond). I was so impressed with the superb graphics that I dug deep into my pockets, got out the readies, and offered the money to a friendly Robtek person. 'Sorry mate! It's not out yet, that's just a few demo screens'. So I earmarked it for a possible purchase when (and if) it actually saw the light of day. I didn't realise at the time what a lucky escape I'd had.

Dizzy Wizard is, I suppose, a sort of Marble Madness clone. It features a kind of black and white chequered spinning top, rotating wildly, roaming around a multi-level 'maze'. By careful manipulation of either the mouse, a joystick or the keyboard, you send it spinning up or down ramps and platforms. The game features superb vertical scrolling, with the screen scrolling up or down with the position of your 'top', keeping it always within view.

The object of the game (and I quote) is to 'pass through the maze as fast as you can in order to light the fire'. This is about all the information you are given. In fact this lack of documentation is one of the great failings of the game. I know that the intention was to leave all the surprises for your discovery but they've gone too far. For example, it says 'each player who has scored less than 10,000 points after one round, has to do as many penalty points on the machine as are needed to achieve that score'. Eh? Okay, I quickly discovered what the machine actually was, since I was pretty bad at the game and continually failed to achieve the target. But as to how one does 'penalty points on the machine' remains a mystery still.

I liked the fact that you have a range of controls (mouse, joystick etc.) but using the things to make your top go in



the direction you actually want it to is nigh on impossible. There is a slow mode which helps on the tricky catwalks and corners but it was still too uncontrollable. In fact my initial curiosity and excitement soon melted away to pure frustration as I didn't know what I was supposed to be doing, and even if I did, I couldn't make the damn thing do it!

It seems that up to three people can play the game simultaneously. There are three tops available and in single-player mode the other two are designated as 'robots', i.e. computer controlled. It seems that you could set one to the mouse, one to the joystick and the last to, say, the keyboard. The instructions make no direct reference to this. Why not? There are a plethora of nasties to get in your way. Usually this takes the form of three bouncing balls. However, interaction with any of these, by hitting them accidentally, seems to produce no effect. The only problems I had getting down 'to light the fire' were caused by my inability to master the truly appalling directional controls. Every now and then a die (dice?) gets in your way. I presume that if you touch this the effect will be random: seems logical no? NO? Every die I touched had an effect for sure, but I just don't know what. They flashed and changed colour, some numbers flashed on the screen, and a couple of times I was presented with what I suppose is a complete view of the maze. Great, it didn't show where me or anything else was and worst of all, I didn't know how to get back into the game. After pressing triggers and doing a tapdance on the keyboard I suddenly re-entered the game but if this was a timeout or a consequence of hitting a key combination I will never know!

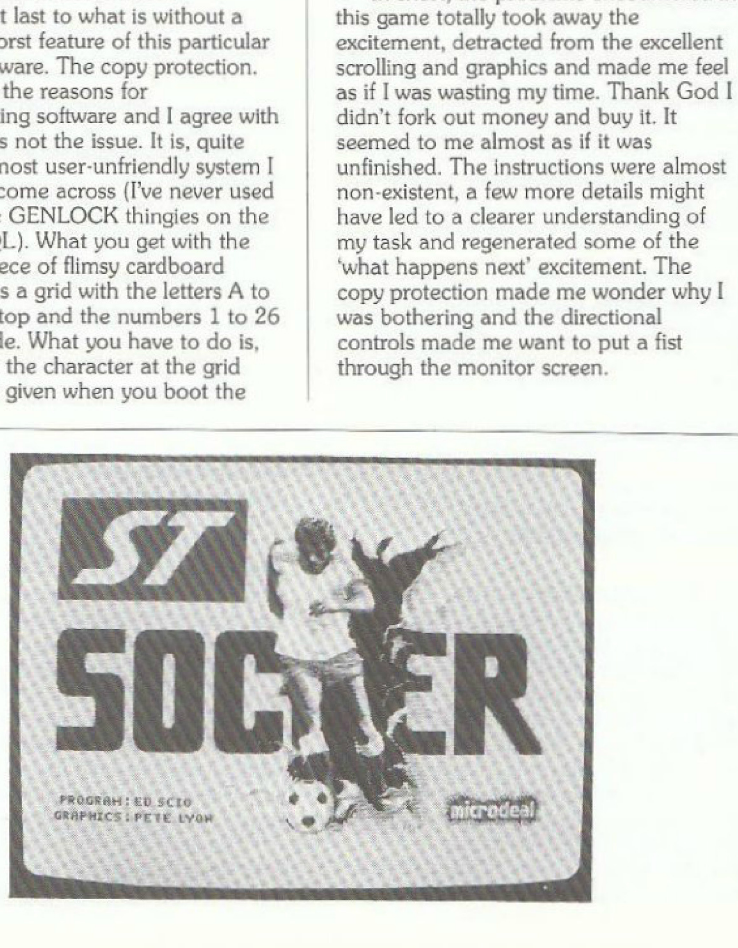
I come at last to what is without a doubt the worst feature of this particular piece of software. The copy protection. Now I know the reasons for copy-protecting software and I agree with them. That is not the issue. It is, quite simply, the most user-unfriendly system I have EVER come across (I've never used one of those GENLOCK thingies on the speccie or QL). What you get with the game is a piece of flimsy cardboard divided up as a grid with the letters A to Z along the top and the numbers 1 to 26 down the side. What you have to do is, get this, find the character at the grid co-ordinates given when you boot the game and enter the character to the left of that position. Sounds easy eh? Well it looks like a 17th generation photocopy, it's on red card (presumably to fox normal photocopiers) and the COLUMNS AREN'T STRAIGHT - Arrrggggh!!!!! Apart from the fact that you can hardly see some of the characters, let alone find their true position, I'm sure that after a few months the card would get dog-eared and probably (if your 'computer room' is anything like mine) lost.

In short, the problems encountered in this game totally took away the excitement, detracted from the excellent scrolling and graphics and made me feel as if I was wasting my time. Thank God I didn't fork out money and buy it. It seemed to me almost as if it was unfinished. The instructions were almost non-existent, a few more details might have led to a clearer understanding of my task and regenerated some of the 'what happens next' excitement. The copy protection made me wonder why I was bothering and the directional controls made me want to put a fist through the monitor screen.

International Soccer

From Microdeal
Price £19.95
Review by Bill Dyer

I'm a soccer fan so I knew I would enjoy this game, but I didn't realise just how good it was going to be! On booting up, you are presented with an intro picture and then an Options screen. This shows two players standing next to a pile of footballs, as you move your joystick one of the balls moves away as your pointer. There are several icons over which you can pass your ball in order to change the characteristics of the game. You can have various team formations, i.e. 3-4-3, 4-5-1, etc. You can select



daytime or floodlight play (the players have shadows under them at night), dry or slippery ground, windy or not, length of each half (5, 10, 30 or 45 minutes), and you can even select the colours each team wears.

Two people can play or you can play the computer. The space bar is pressed to start a two person game, but if you are playing the computer you select your difficulty of game by pressing one of the numbers on the top row of the keyboard (1 is easiest, 9 hardest). The teams come out of the tunnel onto the pitch and take their allotted formation. You can see which player is being controlled by a white marker at his feet, to change to a different player you centre your joystick and press the fire button. The game has all the elements you would expect, tackles, throwins, corners, goalkicks, penalties (movement of the joystick

makes the goalie leap or dive to try and save), and kicking the ball (how hard you kick is dependent on how long you hold down the fire button). A nice touch is the electronic scoreboard, which displays the refs whistle, a penalty has been awarded, and a goal has been scored as they happen.

The standard of the graphics is excellent, and so is the accompanying music (you can select four different tunes with the function buttons). I think Microdeal have another winner here. Top marks.

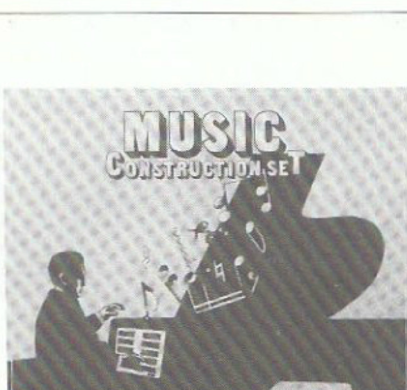
Music Construction Set

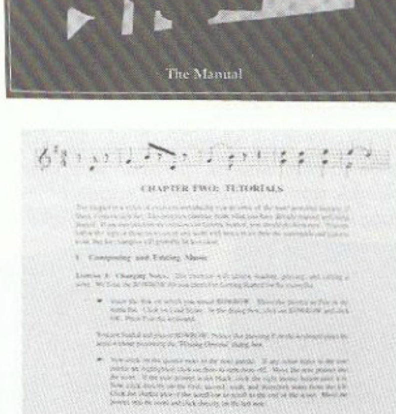
Author: Will Harvey
Distribution: Electronic Arts
Price: £24.95
Reviewer: Glissando
FOR COLOUR MONITORS ONLY

Music Construction Set (MCS) is classified as a STEP TIME EDITING program, similar to the well established Music Studio. This little program will allow you to enter musical scores, either your own compositions or commercial music, into the ST.

Once entered, you may print out the scores, merge DEGAS and NEOCHROME pictures to the music, play the pieces back through the ST's limited sound system or, if you have the necessary facilities, play them back via a suitable MIDI instrument. There is also a Desk Top Accessory which will allow a replay whilst using any GEM program without the requirement to load in the main program. Thus you can listen to your favourite compositions, while you are using some of the more mundane applications, such as Spreadsheets, Word Processing, Data Bases, etc!

The main screen is packed with quite useful features; lots of icon boxes, a blank score sheet, a four octave keyboard, drop down menus with lots of options, icons for notes, embellishments such as triplets and quintuplets, rests, sharps, flats, etc. Notes may be entered in three ways. Firstly, from the ST's

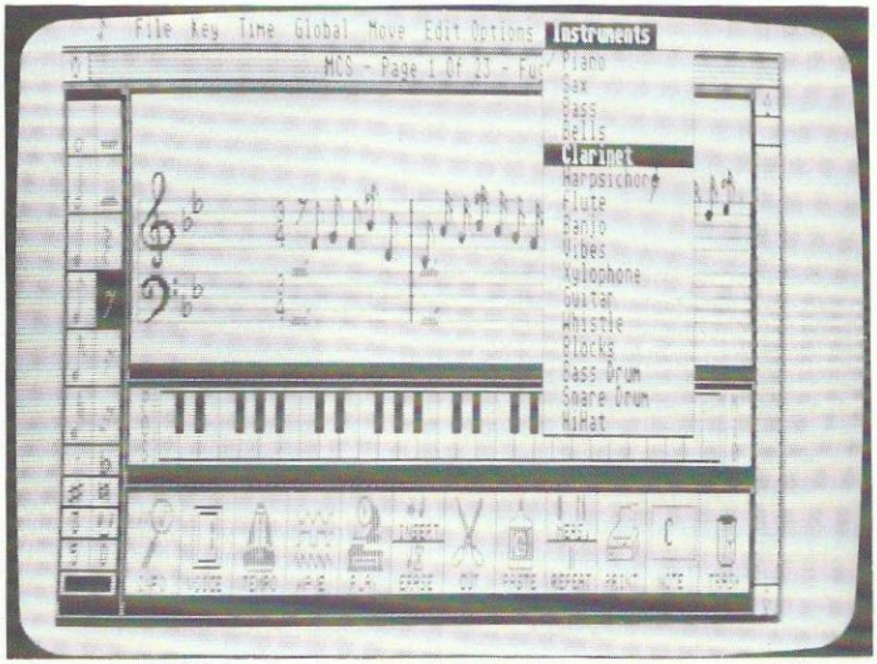



CHAPTER TWO: TUTORIALS

keyboard, secondly, the keyboard on the screen and, finally, from a MIDI keyboard. Note entry is very easy. One simply selects the note type from those available - semi-breve down to the hemidemisemiquaver. In American notation, these equate to the whole note down to the 1/32. The same applies to rest values. The mouse can point directly to the appropriate place on the score, or 'pressing' the screen's keyboard and the note will appear. In the same manner, all the notes are entered. To play back, one simply selects the gramaphone icon. For greater control over the playback, the time signature can be entered at the beginning of the piece, or where they change throughout the piece. The program retains strict control over this feature, whereas MUSIC STUDIO uses it merely as decoration. The key signature may also be entered.

MCS has been advertised as the best step time recorder since the wheel. I am afraid that can be taken with a pinch of salt! There are a number of serious complaints. Let me start with the 'manual'. This is a flimsy pamphlet, no illustrations and no index. What makes matters more painful is the presence on each page of a little piece of decoration. On the left hand page is an extract of music from the treble clef and on the right, the bass. Note the slurs between the notes G and B and top E and D. For the benefit of any non-musical reader, these indicate to the performer that these notes are to be played smoothly. In musical parlance, the passage is to be played LEGGATO. You would think that with this feature advertised on each page, it would be incorporated! Not so. I think this is very misleading. Another valuable feature missing is the complete lack of control of volume. In musical terms these would be expressed from ppp to fff or sF. I would now like to ask the question 'What does your average piece of music sound like with no expression, in terms of playing or volume?' The answer, my friends, is a dirge. In human speech terms, it is not difficult to imagine how somebody would sound if they had no expression in terms of rhythm or volume. Or a piece of writing without any paragraphs, sentences, commas, full-stops, question marks, etc. This does, in fact, exist. Legal documents. Have you read one lately? Perhaps your mortgage, lease, etc. 'Nuff said. To make matters worse, the number of notes permitted is limited to three! This is fine and dandy for strictly ST useage, but with the incorporation of MIDI, provisions should have been made for this user, too. This is easily demonstrated by listening to most of the examples included on the disk, they are pathetic.

Another very annoying feature is the position of the play line. This is an imaginary line at which point the notes are played. When done manually, the eye travels, or scans the score from left to right, from the beginning to the end. In the MCS, the play line is not even on the screen. It is just off the page to the RIGHT of the score! This means that the notes appear AFTER they have been played. Thus it is impossible to play duets - unlike Music Studio, where it is a couple of notes IN from the left. You can see the notes approaching and passing the play line, duets are a doddle!



I would imagine this program is only suitable for the absolute beginner. Any serious work cannot be undertaken unless the piece(s) are very carefully selected.

My verdict: the program needs a great deal more in the way of performance features. The scoring features are adequate, but it is in the performance where the end result appears. The supporting programs on the disk could be useful.

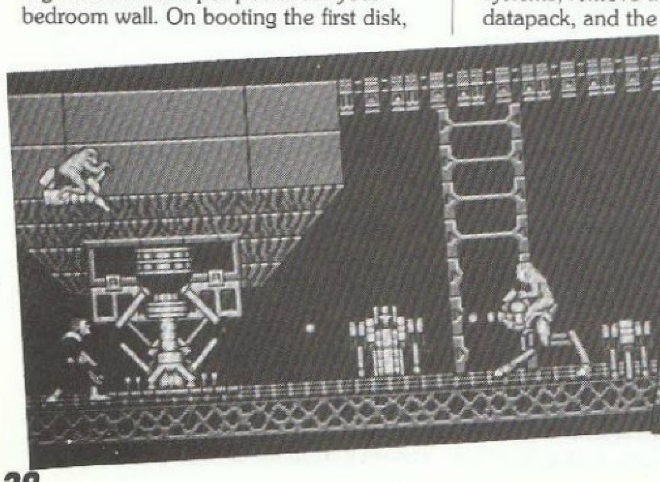
Obliterator

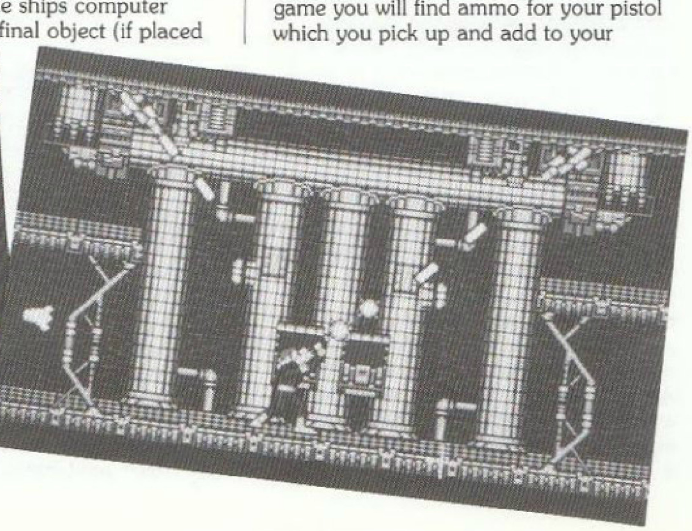
From Psygnosis
Price £24.95
Review by Harry Prince

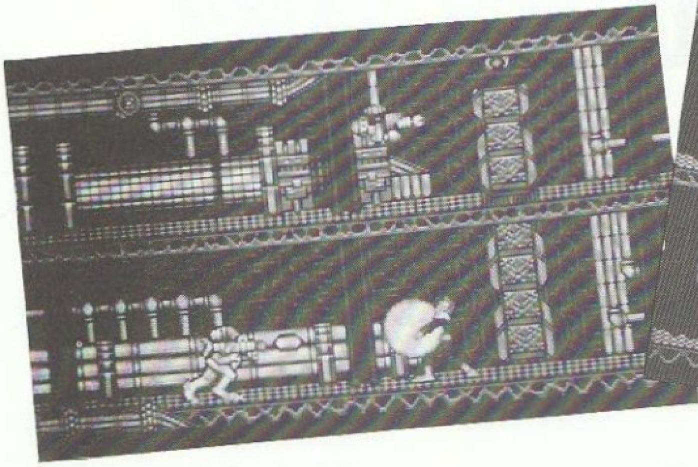
Obliterator is the latest offering from the Liverpool based Psygnosis software house, from whom we have come to expect top quality graphics, and this product is no exception to the rule. Two disks are supplied in the glossy package together with a super poster for your bedroom wall. On booting the first disk, you are presented with some terrific screen pics, (these are some of the best I've seen) and are then invited to insert the second disk (which is kept in the drive during play).

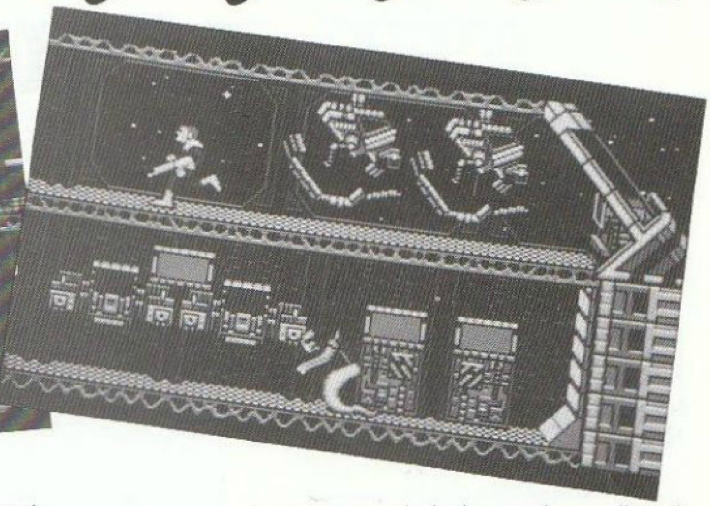
The aim of the game is to board an alien craft and find your way about in order to discover the whereabouts of 5 objects, these objects will then disable the plasma drive of the craft, disable the battle shields, disable the main weapons systems, remove the ships computer datapack, and the final object (if placed correctly) will enable the shuttle system so you can make your escape.

During the time taken to complete your task you will come up against various creatures and contraptions which you must destroy to earn points. Once all 5 tasks have been achieved, your score will start to count back, and you must escape before it reaches zero (so the more points you get the better your chance of escaping). Also during the game you will find ammo for your pistol which you pick up and add to your









arsenal. Other more potent weapons can be found and the ammo for these too. The other weapons are a rifle, a blaster and a bazooka. Your hero (Drak by name) also has a personal energy shield, which deteriates whilst in use, however spares may be found throughout the craft to replenish your pack.

Along the bottom of the screen are the control icons (similar in many ways to Barbarian) which make Drak move, defend himself, shoot, etc. By pressing the space bar a second set of icons appear which give details on the weapons and whether you have collected the five objects or not.

There are over 100 separate locations to search through, all graphically stunning with lots of wierd creatures to fend off. I think Psygnosis have another hit with this one. Very good!

Cambridge LISP

From Metacomco
Price £89.95
Reviewed by Keith Mayhew

Cambridge LISP is a very powerful and extensive implementation of the standard LISP language for the ST. The term 'standard' here is meant rather loosely as there are probably as many different LISPs as there are BASICs!

Common LISP is quickly becoming the adopted standard and Cambridge LISP corresponds quite closely with it, however, there are also some features of Standard LISP incorporated. It should be fairly easy to use code from other LISPs, such as MacLISP and InterLISP, with little change. Although LISP is a language ideally suited to being used to emulate or develop other environments, Cambridge LISP excels in allowing the user to alter its basic syntax. These facilities allow for the ability to alter just about any character in the input, allowing you to change the comment and escape character, etc. by using LISP macros, which are substituted as the parser reads its input.

The library of functions supplied is extensive, their documentation covering a large proportion of the 300 page manual. Full access is provided to all AES, VDI, GEMDOS, BIOS and XBIOS functions allowing this LISP to interact fully with the ST operating system if necessary. As usual, all source code is supplied for these library calls! Access is also provided at a low-level for calling on functions from other languages such as BCPL, so that routines can be coded for maximum speed.

Cambridge LISP is a flexible interpreter allowing more than just the basic integer and list data types. Integers are not of fixed sizes in this LISP and can grow to any length, further there is support for full rational arithmetic, e.g. adding two fractions. Floating point is also implemented to double precision thus making this LISP very strong on arithmetic.

For reasons of efficiency and convenience, the data types of string and vectors are also implemented. The vector facility allows any data types as its elements, whether they are just integers or complex lists or other vectors, which makes this a useful addition for increasing execution speed. Constants are also provided so that the destructive side of LISPs nature can be overidden!

The debugging facilities available enhance this LISP considerably. Functions can be traced and back-tracked, dumps can be made and profiling allows an ideal way of determining which parts of your code are taking most of the execution time. Profiling is an invaluable feature if you are going to develop any large amounts of LISP code. It also provides a good way of learning how not to write hugely inefficient LISP (far to easy!). The error detection facilities of the interpreter are a key to its debugging power, as checks can be automatically made for operations such as trying to access an atom as a list. For speed, these checks can be removed once debugging has been done.

As if this LISP were not good enough, a full-featured compiler is also provided which will compile just about any code the interpreter can execute, producing fast, compact 68000 code. Compiled modules can be called from the interpreter environment increasing the execution speed of code being developed.

To complete the package, Metacomco's standard screen editor, ED, is supplied but has been extended to allow the very necessary feature of matching parentheses! Menu Plus is also supplied but is not really needed as LISP is a self-contained environment allowing the editor to be invoked directly.

If you have been waiting for an advanced implementation of LISP then wait no longer: this is probably the one you need! If you are intending to start learning LISP then this one will be ideal for you and will be more than capable of handling your needs as your abilities grow. However, if you are a beginner, then make sure that LISP is the language you really want to learn as it is not exactly a general purpose language in the sense which you might expect. It is certainly the language behind many successful AI environments and languages: if this interests you then do some initial reading before you take the plunge!

Although 512K bytes is large enough to start using LISP, you must realise that a 1M byte machine is a minimum requirement for serious use. If you have a 512K machine you might want to use this LISP and then upgrade your memory at a later date. Either way, there is unlikely to be a better 'standard' LISP available for your ST in the future.

THE RUSS AI EDITOR FOR THE YAMAHA DX7

Author: Martin Russ
Distribution: Sound on Sound, P.O. Box 30, St. Ives, Cambridge PE17 4XQ
Price: £29.95
Reviewer: Glissando
MONOCHROME MONITORS ONLY

This product is for the Yamaha DX7 series of synthesisers and expansion boxes. In the last edition, I reviewed one of the more traditional Voice Editors for the Yamaha DX7 synthesiser, Laurence Wilkes' DXPERT. There now follows an examination of a brilliantly conceived and new type of Editor.

As you can see from the program title, it uses Artificial Intelligence routines. Totally absent from this program are the means of controlling individual voice parameters. Instead, you manipulate voice characteristics.

One of the features I have mentioned many times is the inherent difficulty the majority of owners have of: a) understanding Frequency Modulation theory as it is applied to the DX7 and b) good voice creation and editing. This program and its manipulation of sound characteristics will open up a new world to many owners of this synthesiser.

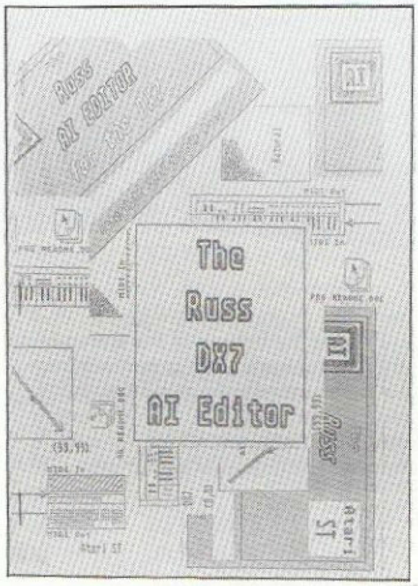
The well used ST expression, 'User Friendly' is an insult! The only action required by the user is to point and click! Another indication of 'User Friendly' is the number of times one has to refer to the manual. I like to, when appraising new programs, run the program first and see how far I can go before I have to turn to the book!

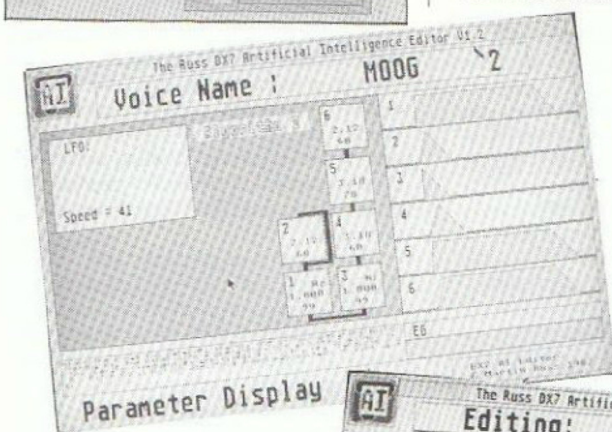
I only had to refer to the manual to check up on any 'hidden' features that I may have missed. This program is great.

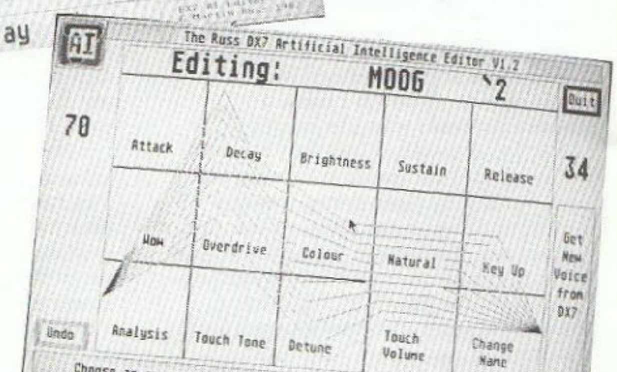
Examine the main screen and you will notice a number of named boxes. All that is required is to click in the box of your choice and the edit is done!

In use, connect the ST to the DX7 via the MIDI leads, if they are not already installed. Run the program. At the prompt, one first has to set the DX7 to System Info Available, select INTERNAL or CARTRIDGE and press the voice number of the voice you wish to edit. I also switch the memory protect to the off position. The program then alerts you to the fact that it has received the voice data you have just transmitted. After a couple of alert boxes, away you go! The boxes, incidentally, only require you to click the left button; no need to hunt for command buttons, etc.

The main command screen has a very attractive, abstract sound waveform as a background and superimposed upon this are the afore-mentioned boxes. Fifteen in all. These are: Attack, Decay, Brightness, Sustain, Release, Wow, Overdrive, Colour, Natural, Key up, Analysis, Touch Tone, Detune, Touch Volume and Change Name. There are also three additional icons, one for QUIT, another for UNDO and another for a NEW VOICE request from the DX7.







Selecting ANALYSIS, a new screen appears displaying numeric and graphic voice characteristics. These include the LFO waveform and frequency, a breakdown, in graphical and verbal terms of the algorithm. Each operator is displayed, showing the output level and frequency. A graphical display of the Eg is also shown.

At the bottom of the screen a comprehensive description of the algorithm is given.

When a voice is loaded for editing, the chosen characteristic and its little box becomes the centre of attention. Each box is divided up into 10,000 points - a matrix of 100 by 100. Therefore, clicking within the box, a reference point is created and it is this information that is analysed and the necessary changes to the parameters are made; in about a second! Very fast.

The top left point has the grid

reference of 00,00 and the bottom right 99,99. Clicking anywhere within the box will produce a greater, or less effect. It depends where you click. If we take the box COLOUR, for example, the 'hot spot' area, where there is a dramatic change in characteristic occurs in the top left hand corner. The exact boundary lies within the area bounded by 35,25 approximately. The author likens these areas to the clutch on a car. From the floor to a point about half an inch from the top of its travel, there is not much of an outward change. Internally, lots of things are happening. Then all of a sudden, the clutch starts to bite and the driver is aware of lots of things happening quite quickly.

Most of the boxes have these little areas of intense activity, their exact area varies from characteristic to characteristic. This is no real problem. Even clicking in the same area again can often produce quite a different sound!

What does need to be done frequently, is SAVE the edit. Sometimes the changes in the edit can be rather excessive and overwhelming! UNDO will take you back to the last edit only, NOT the original. In practice what I do is, initially, copy the voice I wish to edit to position number one AND TWO on the synth. I now transmit the voice to the Editor. As I come across edits that I like, I copy them in turn to number three, four, five, etc. When the bank of thirty two voices is full, I quit out of the program and load DXPERT. Both programs are resident in a RAM disk, created with RDY! Once in DXPERT, I load the whole bank into the system and save it off to a disk - using a suitable name. Quiting DXPERT, I quickly re-load AI and then I can examine and create new sounds from those resident or explore new avenues and I still have the original voice in position number one!

The original edited voices, of course, are still resident in the DX7 and I also have a back-up resident on disk!

Let me briefly conduct you through the boxes. Some of them are very obvious: Change Name, Analysis, Attack, Sustain, etc. Some of them do need a little bit of explanation.

Brightness: One of the voice characteristics used to describe a sound is its brightness. This ranges from being soft, gentle, mellow to raucous and harsh.

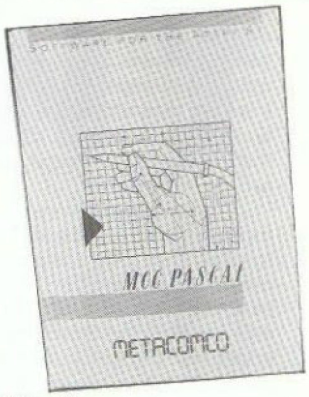
Colour and Natural are related to the composition of the sound. The 'in-word' is timbre. This relates to the overall characteristic quality of the sound produced by, and related to, the overtunes.

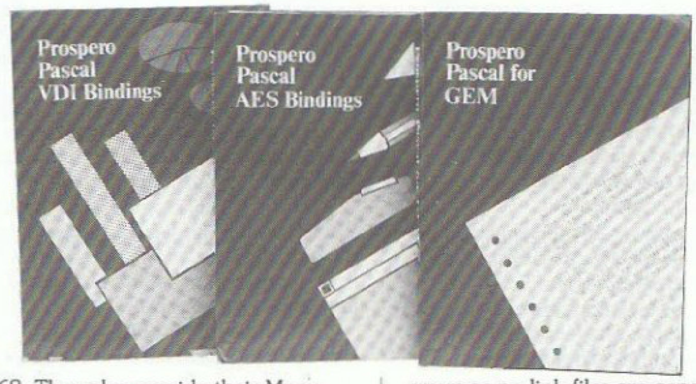
In use, I found the AI Editor refreshingly simple to use and the control over the parameters certainly makes for rapid and efficient voice editing. To start off, keep to the author's recommendation of working along the diagonal top left to bottom right. When you are satisfied that you are familiar with the effects produced from this range, you can move laterally across the box and in a very short time you will be able to anticipate the likely outcome of any region within the box. In a similar manner work through all the boxes again noting the changes. Only then would I recommend working with two, or more boxes.

The AI Editor is very good value for money indeed. Used with DXPERT, together they truly complement each other. Considering the price of currently available voice editors, most of which eminate from the States, the savings are enormous. But one must remember that the latter editors are aimed at the 'PROFESSIONAL' market, where one reads gullible for professional!

Incidentally, Martin Russ has kindly contributed some MIDI and DX7 programs to the library, including a special version of the review program, with reduced features - see the new additions on the ST Library page. Many thanks indeed Martin.

A Comparison of two Pascal Compilers





By Keith Mayhew

Metacomco Pascal £89.95
(Upgrade £29.95)
Prospero Pascal £99.95
(Upgrade £29.95)

Metacomco and Prospero software have both upgraded their Pascal compiler packages and they are both offering advanced and powerful facilities. If you already own either of these packages then upgrades are available at a reasonable cost (see above). These packages provide many more features than their previous versions so you will probably find it well worth the money to upgrade. The extensions to both these compilers allow full access to the ST's operating system so, for instance, fully-fledged GEM applications may be written.

Other extensions provide access to any location of the ST and full bit manipulation, such as AND, OR and shift. These compilers thus offer most of the flexibility of the C language whilst still providing the securities of standard Pascal. This makes Pascal not only a good language for a beginner but also a flexible language which will cope with the needs of more advanced applications. Furthermore, both implementations conform to the I.S.O. standard for Pascal and have been validated, so you can be fairly sure that the compilers do not contain bugs!

The rest of this review will outline the features available in both the packages and will make comparisons between them.

Overview of the Packages

Metacomco's Pascal is a one-pass compiler and is supplied with GST's linker. It can also make object files compatible with Digital Research's Link68. They also provide their Menu Plus command-shell environment for invoking the different tools, which include: a GEM based text editor; a Make program for automating the compilation and linking of multi-file programs; a source-level debugger called Debug Plus and K-Resource - an editor for creating GEM resource files.

This is the same package which is offered with all of Metacomco's languages and provides all the tools you require to create and debug GEM applications. The manual is in the usual, thick, book-style with descriptions of each of the tools, features and extensions of the compiler and full references to all the functions available.

Prospero are offering a similar package which consists of a two-pass compiler and their own linker which is compatible with GST's linker. For the interest of those of you who are thinking of upgrading, you should note that Prospero no longer require you to load a run-time library before you can execute the compiler or any application: so compiled programs are now completely stand-alone.

Also supplied is a good GEM text editor, with 'Word-Star' like key commands. This is a better editor than Metacomco's (which is little more than their old screen-based one) allowing mouse selection of a block of text and direct copying or moving of blocks between windows. Prospero's editor also provides a complete environment from which to compile, link and run programs. This has the advantage that the editor does not have to be re-loaded each time an error occurs.

The compiler and linker are fully integrated into the editor's environment and show their progress in dialogue boxes. If errors occur you only have to click on a button to abort compilation or linking. The compiler can also be invoked to only provide syntax checking, which speeds up the process of finding errors as no link files are generated. After a syntax error has been found the cursor is automatically positioned on the line of the first error ready for correction.

A librarian, which can create linker libraries and extract modules from them, is a useful utility provided with Prospero's package and is something which is missing from Metacomco's. Prospero also provide a cross-reference program which prints out where in a source file identifiers have been declared and used. On the other hand, Prospero do not supply a resource editor which means the extra purchase of, say, K-Resource if you wish to create GEM applications with menus and dialogue boxes. This is the only real omission from Prospero's package, but to some people it may make all the difference. Prospero do compensate slightly by providing additional GEM library functions which can create menus and help to build dialogues, but this is really only suitable for small GEM applications.

A powerful part of both of these packages are their source-level debuggers which will help beginners learn quicker and more experienced programmers to debug their programs quicker. Both allow the user to trace programs with reference to the source line-numbers and to set break-points to stop trace. Variables can be referenced by name and examined or modified. Program output is kept separate from the debug program by having two screens which may be switched over at any time.

Metacomco's 'Debug Plus' allows lower-level access than Prospero's 'Probe', such as disassembly and register or memory dumps. Probe lacks many of these low-level features but offers some very powerful commands to compensate. These include 'watching' variables and printing out their values if they meet the specified conditions and a 'profile' facility to show how many times each individual line is being executed so that you can determine where you can optimise your code if necessary. This obviously slows down the execution of the program but it is worth the penalty to provide easier debugging.

Debug Plus is more flexible than Probe as it allows complex macros to be defined which can be tailored to your exact needs. It also has the added advantage of linking files directly in memory, thus speeding up the whole debugging process. As you can see, both Debug Plus and Probe are excellent and valuable tools - it is left to you to decide between the relative merits of low-level and high-level debugging.

Pascal Extensions and Libraries

Both of the implementations of Pascal support separate compilation of modules so that a program can be split between files for easier management. Prospero have provided a method of accessing routines written either in assembler or their Fortran and Metacomco give you access to their assembler and their C and BCPL languages.

Other features and extensions common to both compilers include: any length variable names; single and double precision real numbers; access to any machine location; bit operators such as AND, OR and shift; random access files; and dynamic strings, i.e. strings of variable length. Prospero provide a slightly better library than Metacomco, allowing allocation of any amount of store using a 'new' call; functions which return the amount of available stack and memory space; a block move; and the following bit manipulations: exclusive or, clear a bit, set a bit, get a bit and toggle a bit. One feature of Metacomco Pascal is the ability to specify number constants in binary and octal as well as the decimal and hexadecimal available from Prospero's.

One significant area of difference between the two implementations is in their support of the system routines. Metacomco provide a function for each system routine and enough documentation to see how to call each one. Propero provide such access only to GEM AES and VDI and parts of GEMDOS. Any other calls you need to make, in particular BIOS and XBIOS, require you to make a Pascal record of the parameters and call one of the interface routines provided. This is not a serious limitation though, as you could still write specific functions for each routine, however, there is no documentation provided on the BIOS or XBIOS so you would need a suitable reference manual. You may also be interested to know that Prospero provide access to the Line-A graphics routines, whereas you would need machine code to do this from Metacomco's.

Prospero more than make up for the lack of BIOS and XBIOS documentation with their excellent manuals. They supply three ring-bound manuals of manageable size, one of these covers the Pascal language and the library functions provided. The other two serve as good reference manuals for GEM's AES and VDI calls. The documentation of these GEM calls in Metacomco's manual consists of little more than a short description of the parameters required for each call, whereas the Prospero manuals provide an overview of each part of the GEM system and small example pieces of code for most calls.

Conclusion

These are both very powerful Pascal packages providing a language environment suitable for beginners whilst allowing the more advanced users most of the flexibility of the C language with the safety of the relatively strict type system of Pascal. Thus a beginner who does not wish to learn C, yet wants a flexible enough language will find one of these two Pascal packages ideal. On the other hand, disappointed users of other high-level languages may find that Pascal is the answer to many of their problems.

ST PROGRAMMING

By Keith Mayhew Part Five

In the last part of this series a simple program was presented which demonstrated many of the practical aspects of a GEM program, in particular the handling of events and the manipulation of windows. We continue this time by looking at how to use the scroll bars on a window, allowing control over the movement around a larger image than the window itself. Having done this, we then move onto a discussion of GEM's object trees which allow complex dialogue boxes to be constructed. Object trees are, however, more versatile and can also be used as the contents of a window, a menu, or to place icons and other images onto the desktop behind the windows.

Scroll Bars

When specifying the components of a window in a 'wind_create' call there are six flags for the scroll bar region. Three are for the vertical scroll bar; their names, as given last time in the 'window.h' include file, are UPARROW, DNARROW and VSLIDE. These are, respectively, the up and down arrow boxes and the slider bar. Their counterparts for the horizontal scroll bar are LFARROW, RTARROW and HSLIDE; the following description will be for the vertical scroll bar, but the horizontal scroll bar is used in an exactly analogous way.

It depends on each application which of the six flags for the scroll bars are specified. Usually a window will have all six, but for a window which is fixed in size it may have none at all. The user's interaction with the scroll bars will generate event messages which the application should then respond to by redrawing the window's contents to reflect the movement and then to actually update the slider's position and size in the scroll bar.

The scroll bar is meant, by convention, to represent the entire size of the document being viewed, while the slider is used to represent the proportion of the document actually visible in the window as well as its position within that document. The term document is used here as it is often text which is viewed in a window but it is intended to refer to any type of image. A document usually has its own line and column units, for text it would tend to be in characters, while for a picture it might be actual pixels. The system defines a slider's position and size by two numbers in the range of 1 to 1000. These values can be obtained or set by calls to 'wind_get' or 'wind_set'.

To get or set the vertical slider position:

```
wind_get(window, WF_VSLIDE,
&position, &dummy, &dummy,
&dummy);
wind_set(window, WF_VSLIDE, position,
0, 0, 0);
```

To get or set the vertical slider size:

```
wind_get(window, WF_VSLSIZE, &size,
&dummy, &dummy, &dummy);
wind_set(window, WF_VSLSIZE, size, 0, 0,
0);
```

Note that as 'wind_get' and 'wind_set' are general routines and always expect six parameters. As we do not use the last three parameters for these calls the address of a 'dummy' (i.e. unused) variable must be passed to 'wind_get' and three dummy values must be passed to 'wind_set'. The symbols WF_VSLIDE and WF_VSLSIZE and their associated WF_HSLIDE and WF_HSLSIZE were also defined in the 'window.h' include file last time.

A slider position of 1 will set it to the top or left of the scroll bar for the vertical and horizontal scroll bars respectively. A value of 1000 sets the slider to the extreme other end of its travel. A slider size of 1000 makes the slider occupy the entire area of the scroll bar and is used when the entire document is visible within the window. A value of 1 for the slider size is used when the window represents one thousandth of the entire document. As it is possible to have a situation which would require a smaller slider a value of -1 indicates to GEM that the slider should be set to its smallest possible physical size. The calculation of the values for the vertical slider's position and size is given below:

```
slider_size = 1000L * doc_seen / doc_
total;
if (slider_size == 0)
slider_size = -1;

slider_position = (999L * (doc_top_line -
1) / (doc_total - doc_seen)) + 1;
```

The variable 'doc_seen' refers to the number of lines of the document currently visible and 'doc_total' refers to the total number of lines in the document. 'doc_top_line' refers to the line number currently displayed at the top of the window, ranging from 1 to (doc_total - doc_seen + 1). It is important to realise that although the document's sizes and the slider's sizes may be referred to by WORD size variables, the intermediate calculation needs to be performed in LONG arithmetic as the multiplications can overflow a WORD size! Hence the constants have 'L' following them to indicate LONG to the compiler which causes the other values to be automatically converted to LONG sizes.

The above calculations should only be performed if the window is not showing the entire document. If the document is entirely visible in the window, i.e. 'doc_seen > doc_total', then the size and position should be forced to 1000 and 1 respectively. Whenever the window is resized or the document changes length the slider values should be re-calculated and set to reflect the change. In some cases it is desirable to add top and bottom 'margins' to a document so that the window can be positioned past the start or end of the actual document. If this is required then the 'doc_total' value should include these margin sizes when the calculations are performed for the slider.

The scroll bars are most commonly updated in response to the user clicking in some region of the scroll bar. The messages associated with scroll bars, defined in the 'events.h' file last time are: WM_HSLID and WM_VSLID if either the horizontal or vertical sliders were dragged to a new position or WM_ARROWED if the user clicked on the 'page' area or arrows. The WM_HSLID and WM_VSLID messages have the new slider position in the fifth WORD of the message buffer. This value is not exactly the same as that used for setting the position as it ranges from 0 to 1000. To calculate the new 'doc_top_line' you can use the following, where 'new_position' is the value returned from the message event:

```
doc_top_line = ((LONG)(doc_total - doc_
seen) * new_position / 1000) + 1;
```

Once the new top line has been calculated the slider needs to be set accordingly, as described above, and the image will be redrawn to reflect the change. Note that any user interaction with the scroll bars causes changes only in the position values so the sizes do not need to be re-calculated.

The WM_ARROWED message is received in any one of eight possible cases, the actual case is determined by the fifth WORD of the message buffer as follows:

0 — Page up.
1 — Page down.
2 — Line up.
3 — Line down.
4 — Page left.
5 — Page right.
6 — Column left.
7 — Column right.

A 'page', by convention, refers to the size of the window so that if 'page up' was selected (by clicking in the grey area above the vertical slider) the page immediately before that currently displayed will be drawn in the window. In fact a page is usually slightly less than exactly one display page so that there are one or two lines of continuity between the displayed pages. A 'line' refers to one document unit in height and a 'column' refers to one document unit in width; these are the amount of movement used when the user clicks on an arrow.

The new top document line is easily calculated for a 'WM_ARROWED' message as it is implicitly known what the movement is in document units: for a page it is 'doc_seen' and for a line it is 1. Thus all that is necessary is to add or subtract from the top line and ensure that the new value is in the bounds allowed, i.e. it is in the range 1 to (doc_total - doc_seen + 1).

That completes the description of how the vertical slider is handled, the horizontal slider, as already mentioned, is handled in an exactly analogous way. It has its own variables for 'doc_seen' and 'doc_total' and keeps track of the left most column displayed.

The only window routine which has not been mentioned at this stage is the 'wind_calc' call which provides a way of obtaining a window's work area from its overall size and vice versa. Although a similar service is provided by 'wind_get', this service differs because it may be used when the window in question is not yet in existence. 'wind_calc' is generally used just before a window is opened to determine the overall sizes required by the 'wind_create' and 'wind_open' calls. This facility is particularly necessary when a window must be constrained to sizes and positions on, say, character boundaries. Obviously the alignment needs to be done based on the work area and we require the overall size to suit such a window before creating it. The routine is also needed if the same process is to be applied to user sizing of the window. In this case the work area is obtained from the supplied overall size, adjusted to the nearest constraints, and the overall size re-calculated from the new work area.

The 'wind_calc' call is used as follows:

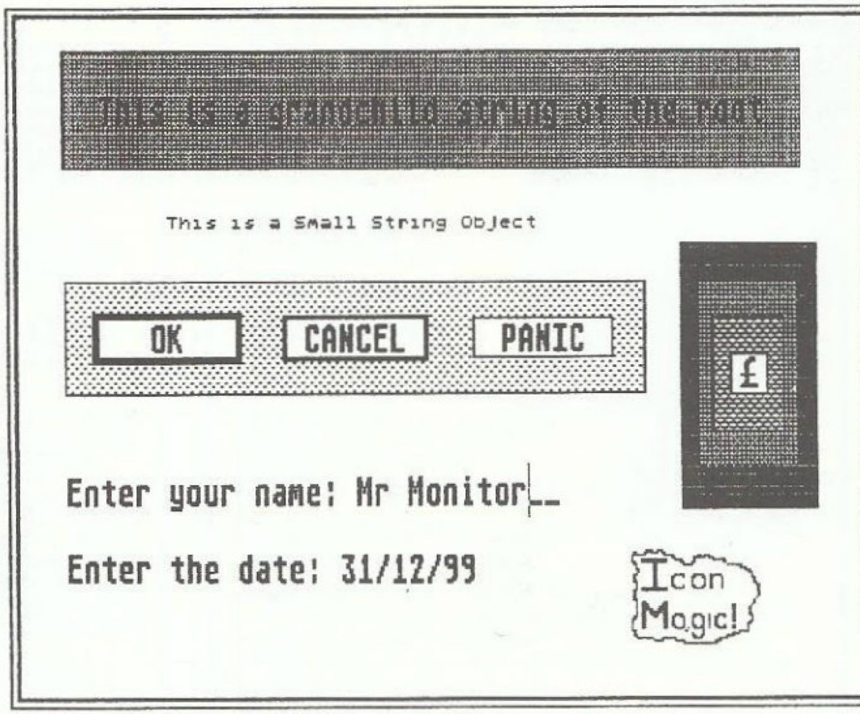


Figure 1.

```
wind_calc(get_inner, parts, in_x, in_y, in_w,
in_h, &out_x, &out_y, &out_w, &out_h);
```

The input area is provided by in_x, in_y, in_w and in_h and the output area is returned into out_x, out_y, out_w and out_h. The 'parts' specify the components of the window as in the 'wind_create' call. 'get_inner' is 1 if we are providing the work area and are receiving the overall area and 0 for the vice versa situation.

Object Trees and Resource Files

We now start an introduction into the exciting world of object trees and resource files. We will develop the principles of these this time and explore more advanced aspects next time.

Most GEM programs have one or several resource files associated with them with the extension of '.RSC'. A resource file consists of one or more object trees held in a special format. When a program loads its resource file into memory, via the AES call 'rsrc_load', the object trees are converted into an internal form. This process, among other things, involves changing from character co-ordinates into resolution dependent pixel co-ordinates.

An 'object tree' is a term used to describe the hierarchical structure of graphical images. Each object in the hierarchy represents a simple graphical image such as the outline of a box, a string, a button, or a user defined icon.

Each dialogue which you interact with is constructed from a hierarchy, or tree, of these basic objects. At the 'root', or base, of the tree is usually a box with a solid background and a highlighted outline. This root object may have many 'children' objects which are drawn inside it and these children can have further children and so on. A property which should not be broken by this structure is that a parent completely encloses all of its children, however, the children do not have to be disjointed and can even completely 'hide' one another.

Children objects in a typical dialogue box are strings of text, buttons, and editable fields which allow information to be entered or changed by the user. It is not that common for there to be any grandchildren of the root box visible in the dialogue. However, in certain cases, some objects are grouped into a parent box which has been 'hidden' by not giving a visible outline. Although this may not at first sound useful it provides an easy way to associate objects for some internal purpose. The most common reason for doing this is to provide 'radio' buttons where clicking on one button causes the other associated buttons to 'pop out', i.e. to be deselected, just as when selecting between bands on some radios, hence their name.

The tree structure of objects is represented internally by an array of structures, one for each object defined. An object structure contains three fields which contain the indices into the array of some of its relatives. One field links an object to a 'sibling' object, i.e. one that has the same parent, and the other two link it to its first and last children. The index of the root object is always zero as it is the very first object in the tree. If an index for a sibling or child is -1 it indicates that the object does not have a sibling or child object, respectively. The only exception to these rules is that the sibling index of the last child of an object points back to its parent. In the case of

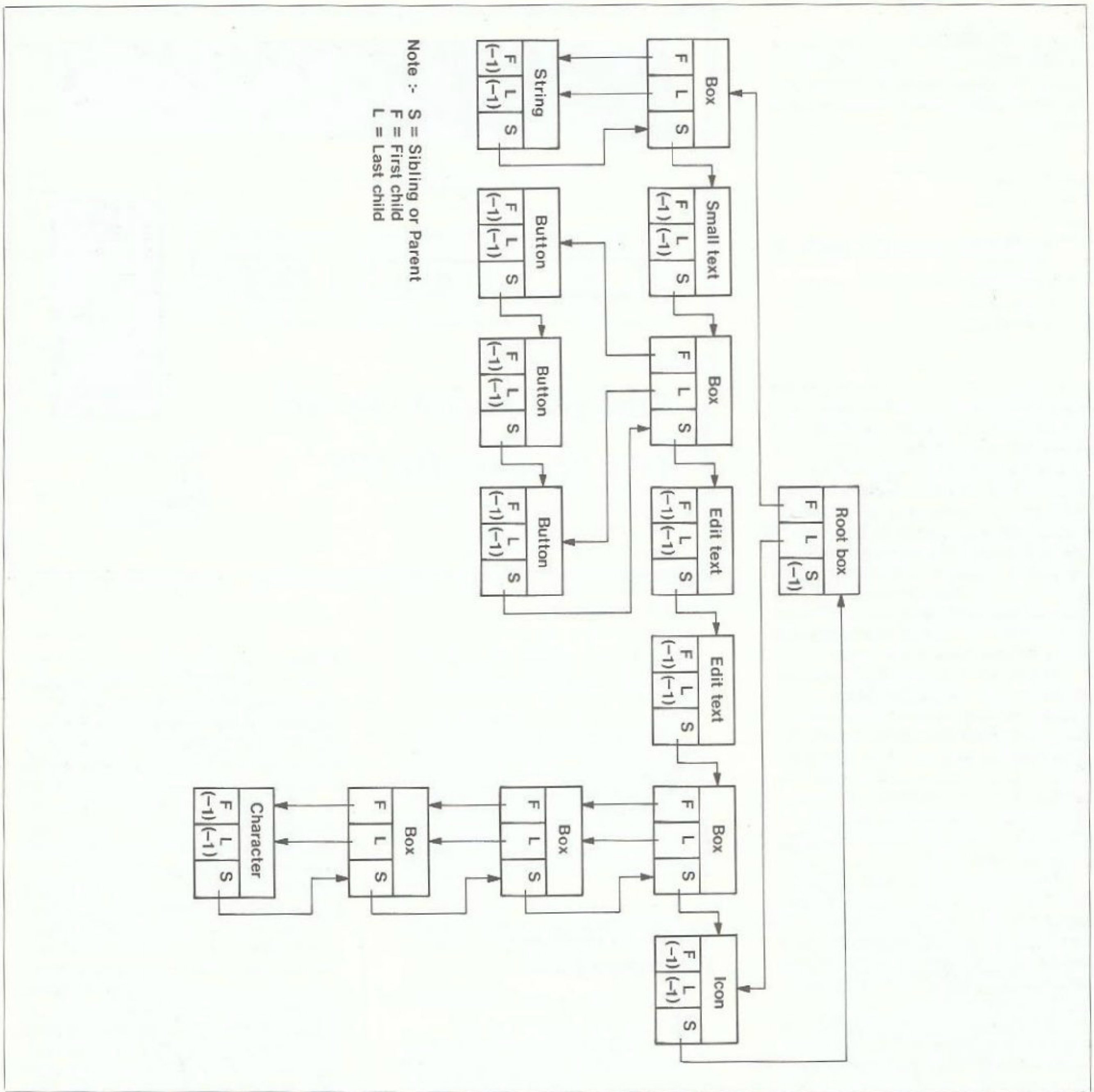


Figure 2.

the root object, its sibling index is always set to -1 as it has no siblings and no parent object. Every other object in a tree should always have a sibling or parent.

Figure 1 is a dialogue box which illustrates many of the different types of object available. Figure 2 is a diagram of the tree structure for this dialogue showing the links between the objects and their types. This tree would be held internally as an array of 15 object structures with WORD size indices for the three links of each. The rest of the information in each object structure describes the object's type and may point to further sub-structures, such as an image, or ordinary text strings.

Each object has four fields for its x, y, width and height. The x and y values for the root are relative to the screen, but for every other object their x and y values are offsets from their parent's x and y co-ordinates. Having each object's position relative to its parent is useful because changing one object's position will move all of its children with it.

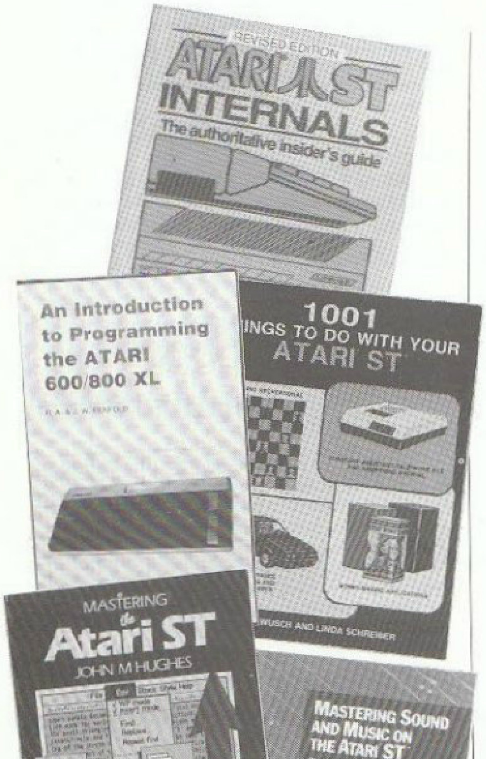
With a tree set up in memory, an 'objc_draw' call will then draw the image on the screen. If we wish the user to interact with the 'form' then a 'from_do' call will handle the interaction so that text strings can be edited and buttons selected. When an 'exit' button is selected the 'form_do' call returns to the user and the information, if necessary, can be extracted.

Next Time

Having had an introduction to object trees we will look in more detail at what constitutes an object and the code necessary to handle them next time. In the meantime, unless you already have a resource construction set, it would be wise if you invested in, say, the K-Resource program. A resource construction set, RCS, is a program which makes the development of object trees easy by handling the objects at their visual level. The RCS writes the '.RSC' files which are then suitable for reading by a program. By separating the object trees from the program itself it makes the trees more easy to maintain as well.

Until then, keep on experimenting, and if you have any problems write to me explaining the trouble. Alternatively, you could let me know what you would like covered next in this series or a proposal for another short series of articles on some particular topic.

Monitor Bookshop



Introducing a new service for Monitor readers, the Monitor Bookshop! Selected titles will be available directly to you by mail. All you have to do is send a cheque or postal order, made payable to 'The U.K. Atari Computer Owners Club', to P.O. Box 3, Rayleigh, Essex SS6 8LR, and state which book(s) you require, together with your name and address. Listed below are the current books available, new additions will be shown as they are available. Prices shown include postage and packing.

GFA Basic - Advanced Programming

This book, by the author of GFA Basic, is the definitive work on this popular implementation. Its chapters take the reader through the steps involved in using the language to the full. Program optimisation and techniques of re-design for greater efficiency in displays and disk access are examined. Graphics including SETCOLOR, clipping and raster commands as well as flicker-free graphics and image storage are covered. Other features include the development of a fully-flexible dialog box, consideration of sound control along with, among other things, recursion, the EXEC command, the creation of fonts and the development of a microscope function. The book comes with a disk of programs from the text and additional examples.

Price £19.50
Order Code MON 01

Mastering Sound and Music on the Atari ST

This book is packed with theory, good programming examples, software/hardware appraisal and much more. Subjects covered include: Acoustics, Making music, Sound programming, Polyphonic music, Sound and music effects, and MIDI applications. Three appendices cover Books, Magazines and Glossary. Sybex have attained a good reputation for quality books, this is no exception.

Price £16.95
Order Code MON 02

Atari ST Internals

This 470 page book goes a long way into describing the functions of all the important components that are found in the ST. There are over 30 illustrations to help clarify the text. The book covers three main topics: the integrated circuits, the interfaces and the ST operating system. Additionally, there is an extensive commented listing of the BIOS. This book is a valuable guide to the student, programmer, novice and expert alike.

Price £14.95
Order Code MON 03

Mastering the Atari ST

The aim of this book is to give an insight to newcommers, in the main, of some of the common applications that the ST can handle in the home and business world. These are word processors, databases and spreadsheets. The software chosen as examples are First Word Plus, LaserBase and VIP Professional. This book is an outstanding buy, it is a very good tutorial.

Price £10.95
Order Code MON 04

1001 Things to do with your Atari ST

This book is a wealth of ideas for uses for your ST. It covers a multitude of subjects including: Personal applications, Business and financial applications, Mathematical and statistical calculations, Technical and scientific applications, Educational and hobby applications, Control and peripheral applications, and much more. Read this book and you'll know the answer to that age old question, 'What can you actually do with a computer?'.

Price £11.60
Order Code MON 05

Computes! Technical Reference Guide

Two superb books from Compute! Volume 1 covers the VDI and Volume 2 explains all about the AES. Both books are filled from cover to cover with vital information and are superbly written. They are written so clearly that they can be recommended for beginner and expert alike.

Volume 1 - The VDI
Price £18.95
Order Code MON 06

Volume 2 - The AES
Price £18.95
Order Code MON 07

An Introduction to Programming the Atari 600/800XL

This book takes the reader step by step, from the fundamentals of Basic on to topics such as animated graphics, variables and arrays, sound generation, strings, etc. A good, economically priced book for beginners.

Price £1.95
Order Code MON 08

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.

Number 18.

Includes: CIO commands and how to use them. Basic checker program to give error messages. Program for 130XE owners to display disk directories on boot-up. Amaurote, Nightmares, Music Matrix, Storm, and a mouse for the XL/XE are reviewed. ST section includes: Useful assembler routines. GEM applications in C including AES windows. Reviews of the Waddington 32 track MIDI sequencer, Enduro Racer, Super Sprint, DXpert V1.4, Mailshot Plus, Chessbase, Lattice C V3.04, Trauma, Rampage and Skyrider.

8 BIT 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.

Monitor Disk 19.

Includes: Oh Damn! an excellent board type game with good graphics and high playability. Searcher is a Turbo Basic program allowing search and replace functions. Programs for use with the Thermometer project. Program to create a new device handler (B:).

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.