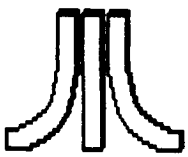
TYNE & WEAR



ATARI 8-BIT USER GROUP

Newsletter of TWAUG

Software
Editorial
Buy & Sell
Hardware

TYNE&WEAR ATARI USER GROUP

Reviews
Help line
Section
Repair Info

Public Domain Library

ISSUE #17

September/October 1995

BRING YOUR EIGHT UP TO DATE with power products from COMPUTER SOFTWARE SERVICES

THE BLACK BOX

The BLACK BOX is an add-on board for the Atari 600XL,800XL and 130XE 8-bit computers. It is a T-shaped board that plugs into the PBI port of the XL computer, or the ECI and cartridge ports of the 130XE. Connectors for both types of computers are built into the BLACK BOX so no adapter boards are necessary. A cartridge port is available on the board itself for 130XE users.

The BLACK BOX provides many unique and useful functions. The four primary functions are:-
* RS-232 serial modem port
* Parallel printer port
* SASI/SCSI hard disk port
* Operating System enhancements

The BLACK BOX is $199.95 for the basic unit, and $249.95 with an onboard 64K printer buffer.
Shipping and Handling extra.

THE BLACK BOX ENHANCER

A must for all BLACK BOX owners. The BLACK BOX ENHANCER is a plug-in module for your BLACK BOX, enhancing the printer functions and adding an instantly available, full featured sector editor!

Installation of the BLACK BOX ENHANCER requires one simple solder connection. Only $49.95 plus shipping/handling.

THE FLOPPY BOARD

Our latest and greatest product. The FLOPPY BOARD is an add-on expansion board for the BLACK BOX interface. It allows the use of the same inexpensive floppy drive mechanisms used in IBM computers. The FLOPPY BOARD is the first floppy drive interface to support "high density" floppy drive mechanisms in either 5.25 inch or 3.5 inch. Built into the FLOPPY BOARD are our BLACK BOX ENHANCER and a version of our SUPER ARCHIVER to allow copying of protected disks for 3.5 inch format. Included with the FLOPPY BOARD is our program to read and write to IBM or ST formatted disks. This makes the FLOPPY BOARD the best way to transfer files to and from your 8-bit.

The FLOPPY BOARD is only $149.95 plus shipping & handling.

THE MULTIPLEXER

This device brings the power and flexibility of larger systems to your 8-bit. The Multiplexer is a collection of cartridge interface boards that allow up to 8 Ataris to read and write to the same drives (typically a hard disk), access the same printer(s), and talk to each other. It is the first practical networking system for the Atari 8-bit computer.

One "master" computer (any 8-bit) is equipped with the master Multiplexer interface. Then up to 8 "slave" computers hook up to this master, each having their own slave interface.

The "common" peripherals (things that are to be shared) are connected to the master. On each slave, all disk and printer I/O is routed through the master, so no extra disk drives are needed.

The Multiplexer sells for $199.95 for a master and two slave units with cable. Additional slave units are $89.95 each, plus shipping/handling.

THE SUPER ARCHIVER II

The SUPER ARCHIVER II edits and copies all enhanced density programs plus retains all the features of the SUPER ARCHIVER.

The SUPER ARCHIVER II is only $99.95 plus shipping & handling. NOTICE: if you already have THE SUPER ARCHIVER you may upgrade to S.A.II for only $29.95 plus shipping/handling. Software only.

THE BIT WRITER

The Super Archiver BIT WRITER is capable of duplicating even the "uncopyable" Electronic Arts and Synapse Syn-series, which employ 34 full sector tracks. The BIT WRITER must be used with the SUPER ARCHIVER

The BIT WRITER is only $79.95 plus shipping/handling.

THE ULTRA SPEED PLUS OS

The Operating System that should be in every XL/XE computer! The Ultra Speed Plus puts unbelievable speed and convenience at your fingertips.

Use any DOS to place Ultra Speed formats on your disks (with XF551 or modified 1050 drives), reading and writing at this speed with most programs. This high speed mode can be turned off for maximum compatibility.

Four simple solder connections are required for installation if your machine has a socketed OS ROM. The Ultra Speed OS is only $69.95 plus shipping/handling.

For more information on these and other 8-bit products:

CONTACT
COMPUTER SOFTWARE SERVICES
PO BOX 17660
ROCHESTER, NEW YORK 14617
USA

ORDERING LINE: (716) 429-5689
FAX: (716) 247-7158
BBS: (716) 247-7157

or contact T.W.A.U.G. we will do our best to help.

EDITORIAL

Who to blame!!!

John Matthewson
David Ewens
Max Gerum

TWAUG Newsletter has been produced entirely on an 8-bit machine. The lay-out of the pages is done using TextPro 4.54 and printed using Daisy-Dot 3 print processor. I am using the 65XE machine upgraded to 1-Meg attached is the Black Box with a 40-Meg Hard Drive. The master copy is printed on a Citizen 120D+.

TWAUG is in no way connected with the ATARI Corporation, except using their product. Atari Logo used on the front cover is for information to our members only and is the Trademark of the ATARI Corporation. Opinons expressed in the TWAUG newsletter is not the Opinon of the TWAUG management but of the various individual authors.

The contribution fees for home and abroad:

HOME	1 COPY	£2.00
-DO-	6 COPIES	£11.00
EUROPE	1 COPY	£2.20
--DO--	6 COPIES	£12.50
ELSEWHERE	1 COPY	£2.50
---DO----	6 COPIES	£14.00

REMINDER:

The book--The Complete and Essential Map--anyone unable to pay the full amount of the cash price can pay in instalments, whenever and whatever you can afford, at no extra charge.

The next issue will be ready by mid-November.

CONTENTS

EDITORIAL	3
MICRO ROUTE TO THE SUN Useful tips for Mini Office Spreadsheet by Andy Doran	4
THE ATARI 8-BIT COMPUTER by Michael Current	7
TIPS FOR PITFALL2 and MOUNTAIN KING	9
ULTRASPEED PLUS OS by Jeff Kyle	10
LETTER SECTION	14
CRACKING THE CODE by Keith Mayhew	15
BOOK REVIEW by Matthew Preston	25
GAMES REVIEWS by Kevin Cooke	27
WLOCZYKIJ REVIEW by Fred Meijer	31
CSS SUPER E-BURNER by Barry Gordon	32
TWAUG ANNOUNCEMENT	35
DISK CONTENT	37
LACE ADVERT	37
OPERATING SYSTEM BLUES by Mark Butler	38
MICRO DISCOUNT	39
ATARI CLASSIC PROGRAMMERS CLUB and CHAOS! COMPUTERS	40

MICRO ROUTE TO THE SUN

Andy Doran find a new use for the Mini Office II spreadsheet

First released in Atari User September 1987.

Most people see spreadsheets as boring programs to use for accounts. This means that lots of interesting uses are overlooked. Let's explore a different use of a spreadsheet using Mini Office II.

If you glance through the glossy pages of a holiday brochure you'll see weather reports and tables which give an indication of the amount of sunshine or rain that a particular resort can expect during the summer months. Such a table can be displayed using a spreadsheet - and with Mini Office II the results can also be shown graphically.

Take the typical British week shown in Figure 1. These figures can easily be entered into the spreadsheet and produce useful figures such as averages.

	MON	TUE	WED	THU	FRI	SAT	SUN
Sun (hours)	8	7	7	8	9	8	9
Rain (inches)	3	2	3	3	1	0	1
Temp (best)	29	28	27	31	32	28	33
Temp (worst)	24	24	25	26	25	26	27

Figure 1: A typical week's weather

Now we'll move on to the program. Once you've loaded it you need to make some changes to the default values given. For instance, there is no point in having two decimal places for numbers. To make these changes select Alter screen display from the main menu and you will move to second menu. The options given are clearly explained on pages 49 and 50 of the Mini Office II manual.

You need to alter the number of decimal places to 0 so simply highlight the option decimal places and enter 0 followed by Return. From now on we will assume that you remembered to press Return as you make an entry. Once this has been done, pressing Escape takes you back to the spreadsheet menu.

Pressing Escape once more will take you to the spreadsheet itself (at present blank). You use the arrow keys to move the cursor around and at the top of the screen you'll notice the status area.

As the cursor is moved this changes to indicate which cell the cursor currently occupies (such as A1 or B6). When data has been entered other sections of the status line such as contents changes to show what is actually in each cell.

The first thing that you need to do is to enter the labels for the spreadsheet - so you know what the values mean later on. To do

MICRO ROUTE TO THE SUN cont.

this we need to move the cursor to cell B1 either by using the cursor keys or the GOTO command - pressing Control+G together gives the prompt Cell in the status area and to move to cell B1 just enter B1.

Now you are at cell B1 you can enter the relevant label. Just type the letters MON and move to cell C1 where you type TUE. Continue until all the days have been entered and you should end up with SUN in cell H1. You now need to enter the labels for Sun, Rain and Temp so use the GOTO function to move to cell A2.

The column width at present is 7 which is not big enough to allow the label Rain (inches) to be entered.

The width of column A (or any other column) can easily be altered by pressing Control+W. You will be prompted for the new column width so enter 13 to allow room for the label. Now you can enter the labels for Sun, Rain and Temp in the same way as you did for the days.

Now your spreadsheet is labelled and ready for you to enter data - except for one thing. You need to enter some formulae to get the information you were originally looking for - averages and so on.

Use the GOTO command to move to cell I1 and label it AVERAGE. Label cells J1 and K1 MAX and MIN respectively and now move to cell I2. This cell will contain a formula to calculate the average daily sunshine (in hours) over the week. The formula is:

I2=(B2+C2+D2+E2+F2+G2+H2)/7

Fortunately Mini Office II allows you to abbreviate this formula using # to stand for sum of. So the formula becomes:

I2=B2#H2)/7

Pressing Control+F allows you to enter the formula. The left hand side of the formula (I2=) appears in the status area and by typing (B2#H2)/7 you will have entered a formula into cell I2. This will read 0 at first as no data has been entered.

The formulae for cells I3 to I5 are are similar to the one you have just entered so rather than enter them again copy them. Make sure the cursor is in cell I2 and press Shift+R.

A message in the status area tells you to move the cursor to the cell you wish to copy to so move the cursor to cell I3 and press Return. The status area now asks if the formula is to be copied absolutely (exactly the same with no changes) or relatively (or similar, but with changes such as B2 becoming B3).

We need to copy the formula relatively so press R for each part of the formula (twice in this case). The new formula I3=(B3#H3)/7 will now be shown in the status area and can be copied to I4 and I5 in a similar way. The formula for MAX is in J2. In J2 you require the formula for the maximum for the numbers

MICRO ROUTE TO THE SUN cont.

between B2 and H2. This is:
J2=>[B2H2].
Enter this in the same way as you entered the formula I2 and copy it into the other cells in the columns (J3 to J5).

The formula for MIN is similar to the formula for MAX, except for the less than (<) sign is used instead of the greater than (>) so enter the last of our formulae into column K starting with K2=<[B2H2].

It is a good idea at this point to save the spreadsheet. This is done by pressing Escape to return to the spreadsheet menu and choosing the Save Spreadsheet option will then cause the computer to prompt:

D1:

At this point enter the name under which this will be saved. It is best to give a descriptive filename so that in future there is no need to guess the contents of the file. Make sure there is a formatted disc in the drive - and not the Mini Office II disc. If you haven't already got one a disc may be formatted by using the option from Mini Office II which is available after a directory. Type WEATHER.V01 (as this is the first version of the spreadsheet). Your spreadsheet will now be saved to disc.

Once you have reached this stage most of the hard work has been done. All that remains is to enter the actual data by moving to the relevant cell and typing the number.

For example move to cell B2 and type 8 Return. When you have entered all the data, save the spreadsheet again and you're then ready to print.

From the main menu choose Print Spreadsheet to move to the print options. You'll have noticed that your spreadsheet has empty rows and columns at the bottom and right-hand sides. It is wasting time printing the whole thing so choose the option Print Part Spreadsheet and enter the following:

Rows:1#5
Columns:A#K

The result will be that only the relevant part of the spreadsheet is printed and when this is completed you will be returned to the print spreadsheet menu. Press Escape to return to the spreadsheet menu and you're ready to save graphics data.

Next month we'll look at how to present this small spreadsheet pictorally. Until then, create some spreadsheets and save them.

T.W.A.U.G.
P.O.BOX No.8
WALLSEND
TYNE & WEAR
NE28 6DQ

THE ATARI 8-BIT COMPUTER

Michael Current

It was 1979 when Atari, Inc. entered the home computer market with the introduction of the 800 and the 400 computer systems. These MOS Technology 6502-based systems run at a clock speed of 1.79 MHz, offering 256 colors displayable simultaneously, up to 320x192 graphics resolution and up to 40x24 text resolution in 11 graphics modes and 5 text modes.

Video may be displayed either on a composite video monitor in the case of the 800, or on a standard television for both systems. 4 independent sound voices are available through the audio output of the television or monitor, each with a 3 1/2 octave range, plus there is a built-in speaker for key-click and other programmable sounds. The 800 has a second cartridge port and a full-stroke keyboard, while the 400 has a single cartridge port and a membrane keyboard. Each has 4 serial controller ports and an Atari Serial Input/Output port.

Originally, both the 400 and 800 were sold with 8K RAM, but later most 800's were sold with 48K and 400's with 16K. Each includes the 10K Atari Operating System in ROM.

The introduction of the 1200XL in 1982 marks the single largest advance in the 8-bit Atari system. The 1200XL runs most software and hardware designed for the 800 and 400, but now runs a slightly more advanced 6502C microprocessor, and includes a full 64K RAM. The single cartridge and monitor ports remain, along with 2 controller ports. In addition, the 1200XL includes 4 programmable Function keys and a Help key, built-in diagnostic and graphics demonstration programs, and probably the favorite keyboard of any 8-bit Atari computer. Clicks previously outputted through the built-in speaker are now heard from the television or monitor's speaker. The revised 14K Operating System offers many new features, including an alternate International Character Set.

In 1983 Atari replaced the 1200XL/800/400 line-up with the new 800XL and 600XL. These new machines include most of the features of the 1200XL minus the Function keys and the demo program. But now both the 800XL and 600XL have the Atari BASIC language built-in. In addition, these two systems offer the Parallel Bus Interface, providing direct memory access to the heart of the computer. The 800XL contains 64K RAM while the 600XL has 16K RAM.

The new Atari Corp. delivered on its promise to advance the 8-bit Atari system by replacing the 800XL/600XL with the new 130XE and 65XE in 1985. The 65XE is nearly identical to the 800XL in features, minus the PBI. The 130XE, however, offers 128K RAM, plus a few special new graphics capabilities. In addition, the 130XE replaces the PBI port with the Enhanced Cartridge Interface, continuing the powerful feature of direct memory access.

THE ATARI 8-BIT COMPUTER cont.

In a change of marketing strategy, Atari introduced the new XE Game System in 1987. Despite its label, the XEGS is a true 8-bit Atari computer system. It offers the convenience of a detachable keyboard and built-in Missile Command game, while offering 64K RAM and full compatibility with the 65XE.

What can you do with an 8-bit Atari computer system? Virtually anything you can do with any other type of computer!
Programming? Pascal, C, BASIC, Logo, Pilot, Forth, Lisp, assembler, plus a unique language called Action! are all available.

Word Processing? Try AtariWriter, Letter Perfect, Paperclip, TextPro, Bank Street Writer, 1st EXLent, TurboWord, Cut & Paste, Letter Wizard...

Database? Try TurboBase, TurboFile, Synfile, Data Perfect, MicroFiler, MegaFiler, Homebase, Super Data Base 1-2-3...

Speadsheet? Look at Syncalc, Visicalc, TurboBase, Calc Magic...

Communications? There's Express!, BobTerm, Kermit-65, Omnicom, VT850, Chameleon...

Graphics? Print Shop, Newsroom, Blazing Paddles, Video Title Shop, Virtuoso, Movie Maker, News Station, Publishing Pro, Awardware, Page Designer, ChromaCAD, Rambrandt...

Music? Virtuoso, Music Studio, Music Construction Set, Songwriter, Electronic Drummer, Music Painter, Music Composer, AtariMusic, MIDI-Track...

Alternate Operating Systems? Don't forget the Diamond Graphic Operating System, bringing point-and-click to the 8-bit Atari, and SpartaDOS X, the last word in Disk Operating Systems for ANY 8-bit computer...

Hardware? The XF551 5.25" disk drive stores 360K per disk. There exist kits to convert it to a 720K 3.5" drive...The Multi I/O or the Black Box may be used to connect a hard drive of any imaginable size...The XEP80 may be added to provide a full 80-column text display...The R-Time-8 is a battery-powered clock device... Third-party memory upgrades are available for up to 4 Megabytes of RAM...SIO-2-PC lets an IBM PC act as a file-server for an 8-bit Atari...

Education? Colorful graphics, exciting sounds and full-screen editing give rise to hundreds of quality educational software titles.

Entertainment? The 8-bit Atari has long been famous for thousands of great games.

If you want to do something with a computer, chances are you can do it on an 8-bit Atari computer. While slower than Intel-based systems like the IBM PC and compatables and 68000-based systems like the Macintosh and the Atari STe/TT, the 8-bit Atari is less costly than any of these, is easier to program at the machine level or alter at the

ATARI COMPUTER cont.

hardware level, and has been documented more thoroughly than any of these newer platforms. The relative simplicity in design of the 8-bit Atari means that many people find the system more reliable than the other types of computers.

The software for the 8-bit Atari is sometimes more powerful than on any other platform: some Bulliten Board Systems are run on 8-bit Ataris specifically because the BBS software available is better than that for any other type of computer. There is a reason for the relative quality of software on the 8-bit Atari. People program commercially for PC, Mac, or STe/TT to make money; people program the 8-bit Atari because they want to - they enjoy producing good software. It takes a team of specialized programmers to develop a major piece of software for those other systems; with the 8-bit Atari, a single person has the chance to learn the entire system, thereby developing greater pride in his programming abilities and his final product.

The 8-bit Atari owner can take pride that his computer platform was developed years before the Macintosh, IBM PC, or Atari ST were on the drawing boards, but remains as useful today as it was in 1979.

For these reasons and more, the 8-bit Atari remains a viable alternative in today's computer marketplace.

TIPS FOR PITFALL2 AND MOUNTAIN KING

To get past the bats in Pitfall II, you have to run underneath them when they are moving up. Be prepared to become frustrated. (But it is possible to get past all the nasties. I have cleared the first level.)

As for Mountain King: (What a great game! It was one of the first that I ever got for my Atari!)

Basically, you are trying to escape the dungeon with the crown. You may have noticed that there is a chamber at the bottom of the maze that is surrounded by fires and has a crown at the top. This is your goal.

Anyway, first you have to gather 1000 diamond points: each cluster of 3 diamonds is worth 10 points. After you get these points (one of the numbers at the bottom tells you how many points you need still) then the Flame Spirit will become available. (The number of the points you need will be replaced with a dancing flame and another number. This number represents how many points you will receive when you find the flame spirit.)
Anyway, once the flame spirit becomes available, wander around (usually near the bottom) until you hear music. Go where the music seems loudest, and use your flashlight to locate the flame spirit. (It is invisible, but will occasionally flash for an instant.) Once you have found the flame spirit, press down on the joystick (this constitutes kneeling).

TIPS FOR PITFALL2 AND MOUNTAIN KING

A static field should be visible around your character.

Now go to the bottom entrance of the crown chamber and kneel for a moment again, until the Skull appears. Now you can climb up the skull and go to get the crown. Climb up the sides and get directly underneath the crown, go up, and kneel again. You should hear a fanfare. Now exit through the bottom entrance, and music should start playing.

You have either two or three repetitions through this song before your crown disappears to get to the very top of the highest mountain at the top of the maze. Go through the flame there, and then you'll be on the next level. Oh, by the way, avoid the bats while you have the crown on the first level, and on all higher levels, when you have either the flame spirit or the crown. The grey bats will take either away. You will also want to avoid the spider which resides on the very bottom level of the maze.

ULTRASPEED PLUS OS

by Jeff Kyle

Do you use an expanded 1050 or an XF551? Do you have more than 64K? Do you use a translator often? If so, or even if not, then you need the UltraSpeed+ OS package from Computer Software Services (CSS), sold for $69.95.

The US+ OS is a collection of 3 OSs for your XL or XE machine. It

ULTRASPEED + OS cont

contains the standard XL/XE OS, the UltraSpeed+ (US+) OS, and the XL-Fix+ OS (XLF+).

The XLF+ OS is an OS that is as compatible as is possible with the 400/800 OS. This lets you use all the programs that would normally require a translator by just flipping to the XLF+ OS. By the OS being ROM, it is also possible to use 400/800 cartridges that normal translators can't handle.

Some commercial programs won't work with a ROM-resident translator, though. An example of this is the older Electronic Arts programs. However, by booting up and holding down SHIFT, the OS will be copied to RAM as if it were a standard disk-based translator, allowing these programs to run also.

Also included in the XLF+ OS is a built-in Mach Ten Menu and writer. This is a program like the public domain Fenders that will load and run binary files from disk. When you boot up with SELECT held down, you'll enter it. It will let you load any of the first 22 files on a disk, get a directory from any drive, and write out a shorter form of Mach Ten menu to the disk. The shorter version will only let you load files from drive one.

The OS also reverses the standard use of OPTION, in other words, to enable Basic, hold down OPTION, to disable Basic, don't hold down OPTION. It also speeds up the keyboard response.

ULTRASPEED PLUS OS continued

The UltraSpeed+ OS is much more. However, all cassette routines, the 1200XL function key routines, RAM and ROM check at bootup, the international character set, and the relocating handler routines have been erased to make room for more features. You'll probably never need them, but if you do, just switch down to the standard OS.

If you have a high-speed drive, the US+ OS will automatically enable the high speed on 1050s with a Super Archiver, Happy Enhancement, US Doubler, Duplicator, Klone, and Density Doubler, or the unmodified XF551 drive. If you have one of those drives and are not using the high speed, this alone is worth the price of the OS. Now you can use UltraSpeed with any unprotected disk you own, and use it in any DOS you use. This allows you to do things like take an Infocom adventure and copy it to a high-speed formatted disk and run it in high speed, speeding up the disk access by approximately 3X. Once you've used UltraSpeed for a while, you'll never go back.

The OS also gives you flexible command over the high speed. Pressing SELECT-OPTION will disable the high speed, and START-OPTION will re-enable it. Also, holding SHIFT or SELECT when formatting will format the disk in standard slow skew instead of high speed skew.

If you've expanded your XL or XE to 256K or beyond, the US+ OS will automatically configure your RAM as a standard, configurable single or double density disk. The nice thing about this is being able to have a standard RAMdisk, compatible with anything. It's easy to format it and write out DOS to it, so whenever you want DOS, you can boot from RAM. Also, it is set up to use the standard 130XE 128K RAM banks last. What this means is that, if you have 320K or more, you can have a full RAMdisk and run a program that uses 128K (such as Video Blitz, 130XE Koala Viewers, AtariWriter Plus, etc) without worrying about damaging anything in RAM.

As in the XLF+ OS, the US+ OS reverses the OPTION key, speeds up the keyboard response, and darkens the background color. However, now you may change some of these along with doing some of the things the 1200XL owners could do with their function keys with your US+ OS, by using CONTROL-n or SHIFT-CONTROL-n. This is what you can do:CONTROL-8 will lock or unlock your keyboard, in case you don't want people messing with it. CONTROL-9 will toggle the internal Basic. This command only goes into effect upon RESET. CONTROL-0 will restore the normal background colors and slow cursor. SHIFT-CONTROL-0 will turn them back. SHIFT-CONTROL-7 will toggle the disk I/O noise on and off. SHIFT-CONTROL-8 will turn the screen DMA off, speeding up most processes by 30% or so. Any other key will reenable it, preferable SHIFT-CONTROL-A, which causes no character. SHIFT-CONTROL-9 toggles the keyboard click.

ULTRASPEED PLUS OS continued

There are other custom keyboard functions with the US+ OS:one is the modification to press SHIFT-CONTROL-Clear instead of SHIFT or CONTROL-Clear to clear the screen. The only problem is that with Action!, SHIFT-CONTROL-Clear will normally bring you to the left side of the screen. This will no longer work in the OS. If you need it, just flip to the standard XL/XE OS, then switch back after you've used the key.

Also, you may now press CONTROL-4, 5, 6, 7 as well the CONTROL--, =, +, and * to move the cursor up, down, left, and right. By pressing CONTROL-Caps, you can not only use the normal graphics characters, but also use the cursor keys without holding down CONTROL.

By either pressing HELP-RESET or pressing SHIFT-ESC then pressing RESET, you can force a cold start of the computer. This is useful when you've got important files in the RAMdisk and the computer won't let you out from a program.

One powerful feature of the OS is it's ability to reconfigure drives for different numbers and boot from any drive, including RAMdisk. When you hold down START after a RESET (warm or cold start), you'll enter a menu that has lists drive 1-9 and what each is assigned as. Normally, they are all assigned to themselves, except drive 4 which is normally RAMdisk. Say you wanted drive 4 to be drive 1 and drive 1 to be drive 4. You'd press RESET and START. You then press C for configure, enter the original drive (1), then the drive you want it to be (4). Then you repeat the procedure for drive 4. It's that easy!

With this menu, you can also change your RAMdisk number just with R for RAMdisk, then enter whatever drive you want it to be.

Also built into the menu is a small sector copier, mainly for drive to RAMdisk or RAMdisk to drive copying. It will read the density, format the destination in the density, and copy the disk a sector at a time. If you want, you can do normal drive to drive copies, but because of the one at a time approach, it won't be as fast as a standard sector copier.

If you've entered the menu from a coldstart, you may also temporarily boot from any drive, just by pressing the drive number. This will swap the drives. By pressing SHIFT-CONTROL-6, you can "unswap" the drives and restore the drives to the original configuration. This won't reset everything, it just swaps back the drive you booted from. For instance, if you booted from drive 3, you could still use drive 1 if you referred to it as drive 3. Pressing SHIFT-CONTROL-6 will reset it so that drive 3 is drive 3 again and drive 1 is drive 1 again.

Occasionally in the standard OS, an OSS "supercartridge" can be ignored due to it's bank selecting. The US+ OS sends a command to "wake it up" to eliminate this problem.

ULTRASPEED PLUS OS continued

You can also easily install an external switch to swap the RAMdisk and drive 1. With this switch, you can make RAMdisk drive 1 at any time.

Unfortunately, if you use SpartaDOS, the reconfiguring will have no effect due to it's using custom routines, bypassing the OS. Also the Virtuoso player will always go to the "real" drive 1 instead of whatever is configured as drive 1. Also SpartaDOS will not boot from RAMdisk, allow the keyboard functions (without a KEY OFF command), or allow high speed with the XF551 drive. But if you MUST use this DOS, it is easily modifiable with a built-in patch called by a simple USR command that will modify SpartaDOS to allow these things.

The new OS has removed the standard Self-Test routines and put in a RAM check supporting up to two megabytes. This is also the FASTest RAM checker I've ever seen. Also, in the same vein, the US+ OS frees up pin 17 on the PIA for easier one and two meg upgrades. Doing so will disable the self test and the standard XL/XE OS, but it will work.

Before the OS, there was a problem if you turned your machine off and back on very quickly, because of the way the high-density RAM chips retain their memory for a few seconds after the loss of power. With the new OS, this problem has been eliminated.

The US+ OS also supports a separately sold RAMdisk write protect switch which will protect either your RAMdisk memory or all of the extended memory from writes.

The documentation that comes with the OS is satisfactory. It explains all the functions in detail with many examples. It explains the functions and the installation completely. It would be nice if it told more of the technical information, but it is better than most information that comes with other custom OSs.

Installation? I can't really say about the difficulty as I had mine installed by the author. But looking through the instructions, mainly it should be easy on the 800XL. On the XEs, most of the time the OS is not socketed in, but soldered in. If it is soldered in, you must desolder the OS, which requires a bit of soldering skill. Other than that, it is mainly soldering a few wires to some of the chips, plus drilling a 1/4" hole for the 3-position switch to switch between the OSs.

And that's the OS! I have found very few compatibility problems, but when they do arise, it's as easy as switching back to the regular XL/XE OS. Having all the special functions available is handy, as well as being able to boot from RAM.

However, there have been problems when switching from the standard XL/XE OS to the XL Fix+ OS. Because of different locations, switching can occasionally make a program point to something that can screw up your RAMdisk. This is rare, but it is a good idea to back up your RAMdisk before jumping into the XLF+ OS. Other than that, I've had very few problems with the OS. It works as stated, and the keyboard functions work with most programs.

LETTER SECTION

Dear TWAUG,

Well done again you guys at TWAUG!...another fine example of what a user group newsletter should be like.

Having seen examples of many different newsletters for countless "older" microcomputers, ranging from the TI99/4a to the Amiga 1200, it must be said that your newsletter is of a very high standard indeed.
I cannot really find any problems with the newsletter (or should that be magazine?...44 pages is NOT just a newsletter!) but it would be nice for "non-experts" like myself (who happens to be a very recent convert to the Atari 8-bit) to be able to read articles that presume NO previous technical knowledge....hardware rewiring projects are at best caried out by experienced "techs" such as yourselves.
Is there any possibility of you running a "beginner's BASIC programming" course in the magazine? I understand that you have to cater for the common denominator amongst your members (I presume that the vast majority of your members/ subscribers are experienced Atari hacks!), but sometimes when I look at the more advanced writings covering machine code on the Atari, I may as well be reading Chinese.
But anyway, cheers for the excellent read, and thanks go to David who has helped me obtain the Atari cassette deck! Cheers David.

Yours Christopher Dalby

** PS- For Mr T.L.Bingham, whose address I have lost, I would like to thank him for the Atari BASIC reference manual that he sent me some time ago.**

Reply by Max (Ed.)

Your letter is very much appreciated, thank you, it is always nice to get some feed back.

We regard our newsletter as a newsletter.We do not use expensive DTP programmes for the layout of our missive, neither do we use professional printing firms to produce it.
The project articles published have been requested. A lot of our subscribers are very keen DIY enthusiasts.
You are saying that reading machine code is like reading Chinese to you, to understand machine code you must start from the beginning and not at the end. The Cracking the Code articles have been in TWAUG since issue 1. It is a well written 17 part series and if you start with issue 1 we are sure you would begin to understand it.
TWAUG is an Atari 8-bit only newsletter and we like to publish a wide and varied range of interesting material for all our members.
A long series on BASIC for beginners is already lined up for publication, starting next issue.

CRACKING THE CODE

by Keith Mayhew

Re-printed by M. Gerum

This article first appeared in "The UK ATARI Computer Owners Club" later renamed "MONITOR"

Part 17

In this part we will be looking at the facilities for accessing the disk drive. We will start with the low-level access which treats the disk on a sector basis. After this, we will look at the file structure imposed on the sectors via the disk file handler 'D:'.

LOW-LEVEL DISK ACCESS

The only built-in routines provided for accessing the disk drive are: read a sector; write a sector; and format. Note that there are no routines built into the operating system to access files on the disk - these are provided by DOS which is loaded from a disk at power-up.

These sector-based routines are referred to as the Disk Handler. It is not, however, a CIO device handler and cannot be accessed directly from the CIO. Refer to Figure one in Part 15 of this series (issue 15) to see where the disk handler is situated in the operating system hierarchy.

The disk handler is called via a JSR to its own vector 'DSKINV' at E453 hex. The variables used by the handler, and their locations, in hex, are as folows:

0301 DUNIT Disk drive number.
0302 DCOMND Command byte.
0303 DSTATS Status byte.
0304 DBUFLO Buffer address low.
0305 DBUFHI Buffer address high.
0308 DBYTLO Number of bytes low.
0309 DBYTHI Number of bytes high
030A DAUX1 Sector number low
030B DAUX2 Sector number high.

The disk handler uses SIO, the Serial Input Output utility, to pass the required command on to the disk drive via the 13-pin serial interface. In fact, the above variables belong to SIO's Device Control Block (DCB) which is used when accessing any device on the serial bus, not just the disk drive.

When the disk drive receives the command its processor decodes it and uses its WD1771 floppy disk controller chip to actually perform the operation.

SECTOR READ COMMAND

Reading a single sector of a disk is achieved by the following: set 'DUNIT' to the number of the disk drive you wish to access (drives are numbered from one onward); set 'DCOMND' to 52 hex, the 'get sector' command; set the buffer address, 'DBUFLO' & 'DBUFHI', to point to a 128 byte buffer; lastly, set 'DAUX1' & 'DAUX2' to the sector number you want read (sectors are numbered from 1 onward) and JSR to 'DSKINV'.

On return, the status byte, 'DSTATS', will be set to 1 if the read was successful and the sector's contents will be in your buffer. If the read failed 'DSTATS' will contain an error code

CRACKING THE CODE continued

between 8A hex and 90 hex, excepting 8D hex: refer to Part 14 of this series in issue 14 for their meanings.

SECTOR WRITE COMMAND

To write a single sector on to a disk the variables are set up as for reading except 'DCOMND' is set to 57 hex, the 'write sector' command. After a successful call to the disk handler the buffer's contents will have been written to the specified sector and 1 will be in 'DSTATS'.

The above command writes a sector with verify, i.e. the disk drive re-reads the sector and compares it against what should have been written. There is an alternative write command, 50 hex, which writes without this verification, thus speeding up the writing process. This is obviously less safe as you are not guaranteed that the data was written correctly: it only proves that the disk was readable, i.e. the drive could find the appropriate sector.

Unfortunately, the disk handler in the original operating system does not recognise the command to write without verify. To achieve the operation you have to access SIO directly! This 'bug' may have been fixed on the XL and XE machines.

FORMAT DISK COMMAND

Before a new disk can be used to write information to, a 'format' has to be written on to the disk. The format for a standard single density drive consists of 40 tracks, each with 18 sectors of 128 bytes each. Special 'headers' and 'trailers' are written around each sector. These contain information for the disk controller's own use such as sector number, track number and a 'CRC' code which is used to validate data read from that sector. As you can see, there is more on a disk than just what you can put there! Fortunately all these tiresome details are hidden from us when we read and write sectors.

To write a new format on to a floppy disk the disk handler variables are set up to specify the disk drive number, a pointer to a 128 byte buffer and a command byte value of 21 hex. The handler does not return until the disk drive has completed the operation. When finished, a status value is returned. If the format was successful, a count of the number of 'bad' sectors which could not be formatted is returned in 'DBYTLO' and 'DBYTHI'. Normally this count is zero, however, if errors did occur then a list of the sector numbers which could not be formatted are returned in the buffer, in low-high format, terminated with a pair of FF hex. This allows for a maximum of 63 bad sectors, however, if a disk has that many errors it really ought to be thrown away!

All disk handler commands return further status information in a four byte block called DVSTAT starting at 02EA hex. The most useful information here is just bit 3 of the first byte which, if set, indicates that a write or format failed due to the disk being write protected. Of interest to some might be the second byte

CRACKING THE CODE continued

which holds the full status byte of the disk drive's floppy controller chip: for information on this you will have to find a data sheet on the WD1771.

FILE-LEVEL ACCESS

Although sector access is very flexible it is not a very retrieving general information as it requires us to break our information down into small blocks and remember on what sector numbers it was all stored.

File-level access is provided by a File Management System (FMS) which maintains a directory of all the files stored on a disk. The FMS is a part of DOS which is loaded at power-up and places an entry in the CIO's handler table for the device 'D:'. All the standard CIO commands for opening, closing, reading and writing are supported by the FMS plus several other commands.

OPEN COMMAND

The OPEN command accepts a file name and an optional drive number. The file name is in the same format as you use from DOS and can contain 'wildcards', i.e. '*' which matches any number of changes and '?' which matches any single letter: the FMS will open the first file it finds in the directory which matches the wildcards, if they are present.

Disk files can be opened for read only or write only. If a file already exist when opened for write only then the file will first of all be set to zero length, losing any previous information. If both read and write is specified then the file is not set to zero length and either reading or writing can be done, in any order, except that writing cannot extend beyond the end of the file's existing length, this is referred to as 'append' mode.

To extend an existing file the append flag, bit 0 of 'AUX1', must be set along with the write flag, bit 3, i.e. 9 in 'AUX1'. Any writing operations will add data to the end of the file without affecting what is already stored in the file.

Note that data written to a file can be lost if a close command is not issued when the file is finished with!

Access to the directory information, as displayed by DOS, is also provided for by the open command: 'AUX1' has to be set to 6, i.e. a read with bit 1 also set. The directory can then be read using normal 'get record' commands, each one returning one line of the directory containing the file name and a sector count in ATASCII text ready to be displayed. The last read before end of file will return a line showing the number of sectors available on the disk. These lines, if directly sent to the editor device, will provide a directory looking exactly the same as that obtained by DOS.

The file name used when opening the directory governs which files will be returned on subsequent read operations. 'D:*.*' will match all files and thus allows the whole directory to be displayed. However, if you only wish to display, say .BAS files, then 'D:*.BAS' is required.

DELETE COMMAND

CRACKING THE CODE continued

Files can be deleted from a disk by specifying a CIO command byte of 21 hex. The file name determines which files are deleted and can contain wildcards, e.g. 'D:*.*' will delete all files from the directory.

RENAME COMMAND

Any name in a directory can be renamed by specifying a CIO command byte of 20 hex. The file name specification you supply consists of two names, seperated by a comma. For example: 'D:FILE1.TXT,FILE2.DAT' will rename 'FILE1.TXT' to 'FILE2.DAT'; note that renaming does not affect the contents of a file in any way. Several files can be renamed at once by using wildcards in both names, for example, 'D:*.BAS,*.TXT' will change the extension of all '.BAS' files to '.TXT'.

LOCK and UNLOCK COMMAND

Files may be 'locked' so that attempts to write to them via the FMS will fail, i.e. write, delete and rename commands. Note that a locked file is not protected physically, so a sector of the file can be written to directly by using the disk handler.

The lock command requires a CIO command byte of 23 hex and a filename which can contain wildcards, in which case all matching files will be locked. Note that locked files are shown in the directory listing by a preceeding star. Unlocking file(s) is done by specifying a CIO command byte of 24 hex and an appropriate filename.

NOTE and POINT COMMAND

The 'note' and 'point' commands are useful when accessing files 'randomly', i.e. in a non-sequential manner. The note command can be issued at any time on a CIO IOCB which is already open for access to a disk file. It requires a CIO command byte of 25 hex and no filename.

On return, it 'notes' the position of the NEXT byte to be read or written: 'ICAX3' and 'ICAX4' specify the sector number in low-high format and 'ICAX5' contains a number between 0 and 124 specifying the position of the byte within the sector.

A 'point' command receives the same information as above in 'ICAX3', 'ICAX4' and 'ICAX5' and 'points' the FMS at the specified byte in a file. Any further reads or writes will start at the specified place, regardless of where reading or writing was previously taking place. The CIO command byte for 'point' is 26 hex and doesn't take filename; it is restricted for use only on files which have been opened for 'append', i.e. read and write.

A typical use of note and point is as follows. When a file is being written a note is made before each item is written and the sector and byte offset are held in memory.When the file is closed and re-opened in append mode any item can be directly accessed by issueing a point command on the relevant sector and byte offset for that item and a subsequent read will retrieve the item.

The information held in

CRACKING THE CODE continued

memory for random access to the file can be written to another file so that, when next used, the program can read in the random access information into memory and access the main file again without having to re-build all the information.

STATUS COMMAND

The standard CIO status command 0D hex, can be used with a filename specifier to determine if a file exists or if it is locked. If no file is found in the directory matching the specified name then an error code of AA hex is returned. If the file is locked, A7 is returned as an error code.

FORMAT COMMAND

A disk can be formatted via FMS using the CIO command byte FE hex and a filename consisting of just a drive number, e.g. 'D1:'. This format operation is similar to that provided by the disk handler but writes some information, such as an empty directory, to the sectors on the disk so that it can be used to hold FMS files.

A SECTOR EDITOR

Listing 1 is the assembly language for a simple sector editor. It uses the read and write sector commands of the disk handler and CIO to display the information on the screen via the editor. The sector is displayed in both hex and ATASCII format and can be edited in either form. Listing 2 is a BASIC listing ready to type in and run. The instructions for use are given in the REM statements at the top of the listing.

With the sector editor you can read, edit and write any sector on a disk you like; but be careful not to use it on a disk containing valuable information, as you might lose it forever if you are not sure exactly what you are doing!!!

This is the last part in this CRACKING THE CODE series.

We appreciate the difficulties some readers had in reading the A5 size of this article. We therefore offer an A4 size of CRACKING THE CODE to anyone interested for a small cover charge for postage, paper and photocopying.

Get in touch with TWAUG for more info about the charge.

Listing 1.

```
0100 ;O.S. vectors...
0110 CIOV    =      $E456   ;CIO entry point.
0120 DSKINV  =      $E453   ;Disk handler entry point.
0130 ;CIO IOCB locations...
0140 ICCOM   =      $0342   ;Command byte.
0150 ICBAL   =      $0344   ;Buffer address low.
0160 ICBAH   =      $0345   ;Buffer address high.
0170 ICBLL   =      $0348   ;Buffer length low.
0180 ICBLH   =      $0349   ;Buffer length high.
0190 ICAX1   =      $034A   ;Auxiliary 1.
0200 ICAX2   =      $034B   ;Auxiliary 2.
0210 ;CIO IOCB commands...
0220 COPEN   =      $03     ;Open device.
0230 CGETREC =      $05     ;Get record.
0240 CGETCHR =      $07     ;Get characters.
0250 CPUTREC =      $09     ;Put record.
0260 CPUTCHR =      $0B     ;Put characters.
0270 CCLOSE  =      $0C     ;Close device.
0280 ;Disk handler (SIO) DCB locations...
0290 DUNIT   =      $0301   ;Drive number.
0300 DCOMND  =      $0302   ;Disk command.
0310 DSTATS  =      $0303   ;Disk status.
0320 DBUFLO  =      $0304   ;Disk buffer address low.
0330 DBUFHI  =      $0305   ;Disk buffer address high.
0340 DAUX1   =      $030A   ;Disk sector number low.
0350 DAUX2   =      $030B   ;Disk sector number high.
```

CRACKING THE CODE continued

```
0360 ;Disk handler commands...
0370 DSKGET  =      $52     ;Get sector.
0380 DSKPUTV =      $57     ;Put sector with verfiy.
0390 ;O.S. equates...
0400 LMARGN  =      $52     ;Editor left margin.
0410 ROWCRS  =      $54     ;Cursor row.
0420 COLCRS  =      $55     ;Cursor colum.
0430 CRSINH  =      $02F0   ;Cursor inhibit.
0440 DSPFLG  =      $02FE   ;Display flag.
0450 RD      =      $04     ;Read from IOCB.
0460 WR      =      $08     ;Write to IOCB.
0470 EOL     =      $9B     ;End of line character.
0480 EOF     =      $88     ;End of file error.
0490 CRSUP   =      $1C     ;Editor cursor up character.
0500 CRSDN   =      $1D     ;Editor cursor down character.
0510 CRSLT   =      $1E     ;Editor cursor left character.
0520 CRSRT   =      $1F     ;Editor cursor right character.
0530 CLS     =      $7D     ;Editor clear screen character.
0540 DELLINE =      $9C     ;Editor delete line character.
0550 ;Program equates...
0560 EDIOCB  =      $00     ;IOCB index for editor 'E:'.
0570 KBDIOCB =      $10     ;IOCB index for keyboard 'K:'.
0580 ;Page zero variables...
0590         *=     $CB
0600 TEMP1   *=     *+1     ;Temporary location.
0610 TEMP2   *=     *+1     ;Temporary location.
0620 CURBYTE *=     *+1     ;Current byte being edited.
0630 SECTOR  *=     *+2     ;Sector number low/high.
0640 DRIVE   *=     *+1     ;Drive number.
0650 EDITX   *=     *+1     ;Edit position X.
0660 EDITY   *=     *+1     ;Edit position Y.
0670 EDITDIG *=     *+1     ;Edit left or right digit.
0680 EDITHEX *=     *+1     ;Edit hex or ATASCII.
0690 PROMPT  *=     *+1     ;Command prompt flag.
0700         *=     $6000
0710         PLA    ;Clean stack for entry from BASIC.
0720         LDX    #EDIOCB ;Open editor, in case closed.
0730         LDA    #EFILE&$FF
0740         STA    ICBAL,X
0750         LDA    #EFILE/256
0760         STA    ICBAH,X
0770         LDA    #RD+WR  ;Read and write.
0780         STA    ICAX1,X
0790         JSR    OPEN
0800         LDX    #KBDIOCB ;Open keyboard.
0810         LDA    #KFILE&$FF
0820         STA    ICBAL,X
0830         LDA    #KFILE/256
0840         STA    ICBAH,X
0850         LDA    #RD     ;Read only.
0860         STA    ICAX1,X
0870         JSR    OPEN
0880         LDA    #0      ;Zero left margin.
0890         STA    LMARGN
0900         JSR    CRSOFF  ;Cursor off.
0910         LDX    #EDIOCB
0920         LDA    #EOL
0930         JSR    PUTBYTE ;Blank line.
0940         LDA    #MSTITLE&$FF ;Print title string.
0950         STA    ICBAL,X
0960         LDA    #MSTITLE/256
0970         STA    ICBAH,X
0980         JSR    WRITELN
0990         LDA    #EOL    ;Blank line.
1000         JSR    PUTBYTE
1010         LDA    #MSHEAD&$FF ;Print header sting.
1020         STA    ICBAL,X
1030         LDA    #MSHEAD/256
1040         STA    ICBAH,X
1050         JSR    WRITELN
1060         LDA    #5      ;Print offset numbers...
1070         STA    ROWCRS  ;Set row.
1080         LDA    #0
1090         STA    TEMP1   ;Start offset at zero.
1100 NEXTOFF LDA    #0
1110         STA    COLCRS  ;Set column.
1120         LDA    TEMP1
1130         JSR    PUTHEX  ;Print it in hex.
1140         INC    ROWCRS  ;Next line.
1150         LDA    TEMP1   ;Get offset.
1160         CLC
1170         ADC    #8      ;Next offset.
1180         STA    TEMP1   ;Save it back.
1190         CMP    #$80
1200         BNE    NEXTOFF ;Last one?
1210         JSR    CRSON
1220         LDA    #1      ;Start at sector 1, drive 1.
1230         STA    DRIVE
1240         STA    SECTOR
1250         LDA    #0
1260         STA    SECTOR+1
1270         STA    PROMPT  ;Turn off prompt flag.
1280         JSR    NEWSECT ;Display it.
1290 ;Main loop: handle commands and editing.
1300 COMMAND LDA    PROMPT
1310         BEQ    SHOW    ;No prompt displayed.
1320         LDA    #22
1330         STA    ROWCRS
1340         LDA    #DELLINE ;Delete line character.
1350         LDX    #EDIOCB
1360         JSR    PUTBYTE ;Delete command prompt.
1370         LDA    #0
1380         STA    PROMPT  ;Clear flag.
1390 SHOW    JSR    SHOWPOS ;Show edit position.
1400         LDX    #KBDIOCB ;Get key.
1410         JSR    GETBYTE
1420         CMP    #EOL    ;Return key?
1430         BEQ    USERCMD ;Process user command.
1440         CMP    #CRSLT  ;Cursor left?
1450         BNE    COM1
1460         JMP    LEFT
1470 COM1    CMP    #CRSRT  ;Cursor right?
1480         BNE    COM2
1490         JMP    RIGHT
1500 COM2    CMP    #CRSUP  ;Cursor up?
1510         BNE    COM3
1520         JMP    UP
1530 COM3    CMP    #CRSDN  ;Cursor down?
1540         BNE    COM4
1550         JMP    DOWN
1560 COM4    TAY
1570         LDA    EDITHEX ;Editing hex?
1580         BNE    INSCHR  ;No.
1590         TYA
1600         JSR    GETHEX  ;Convert to hex digit.
1610         CMP    #$FF    ;Valid?
1620         BEQ    COMMAND ;No.
1630 ;Insert a digit.
1640 INSDIG  TAY
1650         LDA    EDITDIG
1660         BNE    INSTAT  ;Right digit.
1670         TYA
```

CRACKING THE CODE continued

```
1680         ASL    A
1690         ASL    A
1700         ASL    A
1710         ASL    A
1720         STA    TEMP1
1730         LDY    CURBYTE
1740         LDA    BUFFER,Y
1750         AND    #$0F
1760         ORA    TEMP1
1770         STA    BUFFER,Y
1780         JMP    EDTBYTE
1790 INSTRT  TYA
1800         AND    #$0F
1810         STA    TEMP1
1820         LDY    CURBYTE
1830         LDA    BUFFER,Y
1840         AND    #$F0
1850         ORA    TEMP1
1860         STA    BUFFER,Y
1870         JMP    EDTBYTE
1880 ;Insert character.
1890 INSCHR  TYA
1900         LDY    CURBYTE
1910         STA    BUFFER,Y
1920         JMP    EDTBYTE
1930 ;Process user command.
1940 USERCMD LDX    #EDIOCB
1950         LDA    #$FF
1960         STA    PROMPT  ;Set prompt flag.
1970         LDA    #0      ;Show prompt.
1980         STA    COLCRS
1990         LDA    #22
2000         STA    ROWCRS
2010         LDA    #MSCMD&$FF
2020         STA    ICBAL,X
2030         LDA    #MSCMD/256
2040         STA    ICBAH,X
2050         JSR    WRITELN
2060         LDA    #MLCMD-1
2070         STA    COLCRS  ;Set cursor poition.
2080         LDA    #22
2090         STA    ROWCRS
2100         LDA    #CRSRT  ;Show cursor.
2110         JSR    PUTBYTE
2120         LDA    #INBUFF&$FF
2130         STA    ICBAL,X
2140         LDA    #INBUFF/256
2150         STA    ICBAH,X
2160         LDA    #INBLEN
2170         STA    ICBLL,X
2180         LDA    #0
2190         STA    ICBLH,X
2200         JSR    READLN  ;Get input.
2210         LDA    INBUFF
2220         CMP    #'+     ;Next sector
2230         BEQ    CNEXT
2240         CMP    #'-     ;Previous sector?
2250         BEQ    CPREV
2260         CMP    #'R     ;Read sector?
2270         BEQ    CREAD
2280         CMP    #'W     ;Write sector?
2290         BEQ    CWRITE
2300         CMP    #'D     ;Change drive?
2310         BEQ    CDRIVE
2320         CMP    #'S     ;Change sector?
2330         BEQ    CSECTOR
2340         CMP    #'Q     ;Quit
2350         BNE    USER2
2360         LDA    #2      ;Set left margin to default.
2370         STA    LMARGN
2380         LDX    #KBDIOCB ;Close keyboard IOCB.
2390         JSR    CLOSE
2400         JSR    CRSON   ;Cursor on.
2410         RTS            ;End of program...
2420 USER2   JMP    COMMAND
2430 ;Next sector command.
2440 CNEXT   INC    SECTOR  ;End to
2450         BNE    CNEXT2
2460         INC    SECTOR+1
2470 CNEXT2  JSR    NEWSECT
2480         JMP    COMMAND
2490 ;Previous sector command.
2500 CPREV   DEC    SECTOR
2510         LDA    SECTOR
2520         CMP    #$FF
2530         BNE    CPREV2
2540         DEC    SECTOR+1
2550 CPREV2  JSR    NEWSECT
2560         JMP    COMMAND
2570 ;Read sector command.
2580 CREAD   JSR    NEWSECT
2590         JMP    COMMAND
2600 ;Write sector command.
2610 CWRITE  JSR    WRSECT
2620         JMP    COMMAND
2630 ;Change drive command.
2640 CDRIVE  JSR    NUMBER  ;Get drive number.
2650         BEQ    CDRV2   ;Not valid...
2660         LDA    TEMP2   ;Low byte.
2670         STA    DRIVE
2680         JSR    NEWSECT
2690 CDRV2   JMP    COMMAND
2700 ;Change sector command.
2710 CSECTOR JSR    NUMBER  ;Get sector number.
2720         BEQ    CSCT2   ;Not valid...
2730         LDA    TEMP2   ;Low byte.
2740         STA    SECTOR
2750         LDA    TEMP1   ;High byte.
2760         STA    SECTOR+1
2770         JSR    NEWSECT
2780 CSCT2   JMP    COMMAND
2790 ;Get number from input buffer.
2800 NUMBER  LDY    #1      ;Next character index.
2810         LDX    #0      ;Number of digits so far.
2820         STX    TEMP1   ;Zero result.
2830         STX    TEMP2
2840 NUM1    LDA    INBUFF,Y ;Get to end of line.
2850         CMP    #EOL
2860         BEQ    NUM2
2870         INY
2880         JMP    NUM1
2890 NUM2    DEY
2900         BEQ    NUM3    ;At start of line.
2910         LDA    INBUFF,Y ;Get character.
2920         JSR    GETHEX  ;Convert hex digit.
2930         CMP    #$FF
2940         BEQ    NUM3    ;Not hex digit.
2950         PHA
2960         TYA            ;Get digit no.
2970         AND    #1
2980         BEQ    NUM4    ;Don't shift if odd.
2990         PLA
3000         ASL    A       ;Shift.
3010         ASL    A
```

CRACKING THE CODE continued

```
3020          ASL     A
3030          ASL     A
3040          PHA
3050 NUM4     TXA             ;Get digit no.
3060          LSR     A       ;Divide by 2.
3070          BNE     NUM5    ;High byte of number.
3080          PLA             ;Or to low byte.
3090          ORA     TEMP2
3100          STA     TEMP2
3110          JMP     NUM6
3120 NUM5     PLA             ;Or to high byte.
3130          ORA     TEMP1
3140          STA     TEMP1
3150 NUM6     INX
3160          CPX     #4
3170          BNE     NUM2    ;Next character.
3180 NUM3     TXA             ;Return zero if no digits.
3190          RTS
3200 ;Cursor left.
3210 LEFT     LDA     EDITHEX ;Editing hex digits?
3220          BNE     LEFT1   ;No.
3230          LDA     EDITDIG ;Toggle digit.
3240          EOR     #1
3250          STA     EDITDIG
3260          BNE     LEFT3
3270          JMP     COMMAND ;End if on same hex number.
3280 LEFT3    DEC     CURBYTE
3290          DEC     EDITX
3300          BMI     LEFT4
3310          JMP     COMMAND ;End if not off left.
3320 LEFT4    INC     EDITHEX ;Move to ATASCII.
3330 LEFT2    LDA     #7      ;Move to last character.
3340          STA     EDITX
3350          LDA     CURBYTE ;Adjust index.
3360          CLC
3370          ADC     #8
3380          STA     CURBYTE
3390          JMP     COMMAND ;Back to command loop.
3400 LEFT1    DEC     CURBYTE
3410          DEC     EDITX
3420          BMI     LEFT5
3430          JMP     COMMAND ;Skip if not off left.
3440 LEFT5    DEC     EDITHEX ;Editing hex.
3450          LDA     #1      ;Right digit.
3460          STA     EDITDIG
3470          JMP     LEFT2   ;Adust index.
3480 ;Cursor right.
3490 RIGHT    LDA     EDITHEX
3500          BNE     RIGHT1  ;Edit ATASCII
3510          LDA     EDITDIG
3520          EOR     #1
3530          STA     EDITDIG
3540          BEQ     RIGHT3
3550          JMP     COMMAND ;Skip if on same number.
3560 RIGHT3   INC     CURBYTE
3570          INC     EDITX
3580          LDA     EDITX
3590          CMP     #8
3600          BEQ     RIGHT4
3610          JMP     COMMAND ;Skip if not off right.
3620 RIGHT4   INC     EDITHEX
3630 RIGHT2   LDA     #0      ;Adjust index.
3640          STA     EDITX
3650          LDA     CURBYTE
3660          SEC
3670          SBC     #8
3680          STA     CURBYTE
3690          JMP     COMMAND
3700 RIGHT1   INC     CURBYTE
3710          INC     EDITX
3720          LDA     EDITX
3730          CMP     #8
3740          BEQ     RIGHT5  ;Correct if off right.
3750          JMP     COMMAND
3760 RIGHT5   DEC     EDITHEX
3770          LDA     #0
3780          STA     EDITDIG
3790          JMP     RIGHT2
3800 ;Cursor up.
3810 UP       LDA     CURBYTE
3820          SEC
3830          SBC     #8
3840          STA     CURBYTE
3850          DEC     EDITY
3860          BMI     UP1
3870          JMP     COMMAND ;Skip if not off top.
3880 UP1      LDA     #15
3890          STA     EDITY
3900          LDA     CURBYTE
3910          CLC
3920          ADC     #$80
3930          STA     CURBYTE
3940          JMP     COMMAND
3950 ;Cursor down.
3960 DOWN     LDA     CURBYTE
3970          CLC
3980          ADC     #8
3990          STA     CURBYTE
4000          INC     EDITY
4010          LDA     EDITY
4020          CMP     #16
4030          BEQ     DOWN1
4040          JMP     COMMAND ;Skip if not off bottom.
4050 DOWN1    LDA     #0
4060          STA     EDITY
4070          LDA     CURBYTE
4080          SEC
4090          SBC     #$80
4100          STA     CURBYTE
4110          JMP     COMMAND
4120 ;New sector: read and display.
4130 NEWSECT  JSR     RDSECT
4140          JSR     CRSOFF
4150          LDA     #1      ;Print drive and sector...
4160          STA     ROWCRS
4170          LDA     #7
4180          STA     COLCRS
4190          LDA     DRIVE
4200          JSR     PUTHEX  ;Drive number.
4210          LDA     #18
4220          STA     COLCRS
4230          LDA     SECTOR+1
4240          JSR     PUTHEX  ;Sector high.
4250          LDA     SECTOR
4260          JSR     PUTHEX  ;Sector low.
4270          LDA     #$FF    ;Display special characters.
4280          STA     DSPFLG
4290          JSR     DISPLAY
4300          LDA     #0      ;Re-enable special characters.
4310          STA     DSPFLG
4320          JSR     CRSOM
4330          LDA     #0      ;Set edit variables.
```

CRACKING THE CODE continued

```
4340        STA     CURBYTE
4350        STA     EDITI
4360        STA     EDITY
4370        STA     EDITDIG
4380        STA     EDITHEX
4390        RTS
4400 ;Byte edited: re-display it.
4410 EDTBYTE JSR    CRSOFF
4420        LDA     #$FF
4430        STA     DSPFLG
4440        JSR     DSPBYTE
4450        LDA     #0
4460        STA     DSPFLG
4470        JSR     CRSON
4480        JMP     RIGHT   ;Move cursor on to next character.
4490 ;Show cursor at edit position.
4500 SHOWPOS LDA    EDITY   ;Get Y position.
4510        CLC
4520        ADC     #5
4530        STA     ROWCRS  ;Set cursor row.
4540        LDA     EDITHEX ;Editing hex or ATASCII?
4550        BNE     ATASCII
4560        LDA     EDITI   ;Get X position.
4570        STA     TEMP1   ;Times 3...
4580        CLC
4590        ADC     TEMP1
4600        ADC     TEMP1
4610        ADC     #3      ;Add offset from left.
4620        ADC     EDITDIG ;Add offset for right digit.
4630        STA     COLCRS  ;Save column.
4640        JMP     SHOW1
4650 ATASCII LDA    EDITI   ;Get X position.
4660        CLC             ;Add offset from left.
4670        ADC     #27
4680        STA     COLCRS  ;Save column
4690 SHOW1  INC     COLCRS  ;Move one right.
4700        JSR     CRSON   ;Enable cursor.
4710        LDA     #CRSLT  ;Send cursor left to editor...
4720        LDX     #EDIOCB
4730        JSR     PUTBYTE ;Cursor now displayed.
4740        RTS
4750 ;Display buffer on screen.
4760 DISPLAY LDA    #0      ;Start at byte 0.
4770        STA     CURBYTE
4780 DSPNEXT JSR    DSPBYTE ;Display it.
4790        LDA     CURBYTE ;Next byte.
4800        CLC
4810        ADC     #1
4820        STA     CURBYTE
4830        CMP     #$80
4840        BNE     DSPNEXT
4850        RTS
4860 ;Display current byte of buffer.
4870 DSPBYTE LDA    CURBYTE
4880        AND     #$F8    ;Get line number.
4890        LSR     A
4900        LSR     A
4910        LSR     A
4920        CLC
4930        ADC     #5
4940        STA     ROWCRS
4950        LDA     CURBYTE
4960        AND     #$07    ;Get column number.
4970        STA     TEMP1
4980        CLC
4990        ADC     TEMP1   ;Times three...
5000        ADC     TEMP1
5010        ADC     #3
5020        STA     COLCRS
5030        LDY     CURBYTE ;Get byte.
5040        LDA     BUFFER,Y
5050        JSR     PUTHEX  ;Show it in hex.
5060        LDA     CURBYTE ;Find column for character.
5070        AND     #$07
5080        CLC
5090        ADC     #27     ;Offset across screen.
5100        STA     COLCRS
5110        LDY     CURBYTE ;Get byte again.
5120        LDA     BUFFER,Y
5130        LDX     #EDIOCB
5140        JSR     PUTBYTE ;Print character.
5150        RTS
5160 ;Convert character to hex digit.
5170 GETHEX  CMP    #'0
5180        BCC     NOTHEX  ;Not valid character.
5190        CMP     #':'    ;Character after 9.
5200        BCS     GETHEX2 ;Not a digit.
5210        SEC
5220        SBC     #'0'    ;Convert to number.
5230        RTS
5240 GETHEX2 CMP    #'A'    ;Upper case letter?
5250        BCC     NOTHEX
5260        CMP     #'G'
5270        BCS     GETHEX3
5280        SEC
5290        SBC     #$37    ;Convert to number.
5300        RTS
5310 GETHEX3 CMP    #'a'    ;Lower case letter?
5320        BCC     NOTHEX
5330        CMP     #'g'
5340        BCS     NOTHEX
5350        SEC
5360        SBC     #$57    ;Convert to number.
5370        RTS
5380 NOTHEX  LDA    #$FF    ;Not a valid hex character.
5390        RTS             ;Return $FF.
5400 ;Print byte in hex.
5410 PUTHEX  PHA            ;Save byte.
5420        LSR     A
5430        LSR     A
5440        LSR     A
5450        LSR     A
5460        JSR     PUTHDIG ;Print high nibble.
5470        PLA
5480        AND     #$0F
5490        JSR     PUTHDIG ;Print low nibble.
5500        RTS
5510 ;Print a single hex digit.
5520 PUTHDIG CMP    #10     ;Is digit nine or less?
5530        BCS     LETTER  ;No.
5540        CLC
5550        ADC     #'0'    ;Turn into ATASCII.
5560        LDX     #EDIOCB
5570        JSR     PUTBYTE
5580        RTS
5590 LETTER  CLC
5600        ADC     #$37    ;Turn into ATASCII.
5610        LDX     #EDIOCB
5620        JSR     PUTBYTE
5630        RTS
5640 ;Open a file.
5650 OPEN    JSR    CLOSE   ;Make sur channel is closed.
5660        LDA     #COPEN
5670        STA     ICCOM,X
```

CRACKING THE CODE continued

```
5680         JSR     CIOV
5690         RTS
5700 ;Close a file.
5710 CLOSE   LDA     #CCLOSE
5720         STA     ICCOM,X
5730         JSR     CIOV
5740         RTS
5750 ;Read up to end of file.
5760 READLN  LDA     #CGETREC
5770         STA     ICCOM,X
5780         JSR     CIOV
5790         RTS
5800 ;Write up to end of line.
5810 WRITELN LDA     #CPUTREC
5820         STA     ICCOM,X
5830         LDA     #$FF    ;Maximum buffer length.
5840         STA     ICBLL,X
5850         STA     ICBLH,X
5860         JSR     CIOV
5870         RTS
5880 ;Get a single byte in accumulator.
5890 GETBYTE LDA     #CGETCHR
5900         STA     ICCOM,X
5910         LDA     #0      ;Zero buffer length.
5920         STA     ICBLL,X
5930         STA     ICBLH,X
5940         JSR     CIOV
5950         RTS
5960 ;Put a single byte from accumulator.
5970 PUTBYTE PHA
5980         LDA     #CPUTCHR
5990         STA     ICCOM,X
6000         LDA     #0      ;Zero buffer length.
6010         STA     ICBLL,X
6020         STA     ICBLH,X
6030         PLA
6040         JSR     CIOV
6050         RTS
6060 ;Turn cursor off.
6070 CRSOFF  LDA     #$FF
6080         STA     CRSINH
6090         RTS
6100 ;Turn cursor on.
6110 CRSON   LDA     #0
6120         STA     CRSINH
6130         RTS
6140 ;Read sector into buffer.
6150 RDSECT  LDA     #DSKGET ;Get sector command.
6160         JSR     SETDSK
6170         JSR     DSKINV  ;Read the sector.
6180         RTS
6190 ;Write sector from buffer.
6200 WRSECT  LDA     #DSKPUTV ;Put with verify command.
6210         JSR     SETDSK
6220         JSR     DSKINV  ;Write the sector.
6230         RTS
6240 ;Set disk handler variables.
6250 SETDSK  STA     DCOMND  ;Set command.
6260         LDA     DRIVE   ;Set drive number.
6270         STA     DUNIT
6280         LDA     #BUFFER&$FF ;Set buffer address.
6290         STA     DBUFLO
6300         LDA     #BUFFER/256
6310         STA     DBUFHI
6320         LDA     SECTOR  ;Set the sector number.
6330         STA     DAUX1
6340         LDA     SECTOR+1
6350         STA     DAUX2
6360         RTS
6370 ;Messages etc.
6380 MSTITLE .BYTE   "Drive: 01 Sector: 0001",EOL
6390 MSHEAD  .BYTE   "   +0 +1 +2 +3 +4 +5 +6 +7 01234567",EOL
6400 MSCMD   .BYTE   "Command?",EOL
6410 MLCMD   =       *-MSCMD ;Length of message.
6420 EFILE   .BYTE   "E:",EOL ;Editor file spec.
6430 KFILE   .BYTE   "K:",EOL ;Keyboard file spec.
6440 ;Buffer for user input.
6450 INBUFF  *=      *+$80
6460 INBLEN  =       *-INBUFF ;Length of buffer.
6470 ;Buffer for sector.
6480 BUFFER  *=      *+$80
```

Listing 2.

```
10 DIM HEX$(16)
20 J=0:START=24576:TRAP 90
30 READ HEX$
40 FOR I=1 TO 15 STEP 2
50 D1=ASC(HEX$(I,I))-48:D2=ASC(HEX$(I+1,I+1))-48
60 NUM=((D1-7*(D1>16))*16+(D2-7*(D2>16)))
70 POKE START+J,NUM:J=J+1:NEXT I
80 GOTO 30
90 I=USR(START)
100 REM Instructions
110 REM ============
120 REM
130 REM Once started, you will see
140 REM sector 1 of drive 1. You can
150 REM edit any byte in hex or ATASCII.
160 REM Pressing return will prompt fo
170 REM a command. You can type any of
 the following:
180 REM 'W' (Write the sector back to
disk)
190 REM 'R' (Read the sector back from
 disk)
200 REM '+' (Read the next sector)
210 REM '-' (Read the previous sector)
220 REM 'D number' (Change to specified drive)
230 REM 'S number' (Read the specified
 sector)
240 REM 'Q' (Quit program)
250 REM
260 REM Note that 'number' for 'D' or
'S'
270 REM commands must be given in hex.
280 REM
1000 DATA 68A200A9A79D4403
1001 DATA A9649D4503A90C9D
1002 DATA 4A0320D563A210A9
1003 DATA AA9D4403A9649D45
1004 DATA 03A9049D4A0320D5
1005 DATA 63A9008552202064
1006 DATA A200A99B201564A9
1007 DATA 639D4403A9649D45
1008 DATA 0320F363A99B2015
1009 DATA 64A97A9D4403A964
1010 DATA 9D450320F363A905
1011 DATA 8554A90085CBA900
1012 DATA 8555A5CB20B063E6
1013 DATA 54A5CB18690085CB
1014 DATA C9B0D0EA202E64A9
1015 DATA 0185D085CEA90085
1016 DATA CF85D520BA62A5D5
1017 DATA F00FA9168554A99C
1018 DATA A200201564A90085
1019 DATA D5200E63A2102004
1020 DATA 64C99BF061C91ED0
```

CRACKING THE CODE cont

```
1021 DATA 034C0262C91FD003
1022 DATA 4C3C62C91CD0034C
1023 DATA 7E62C91DD0034C9A
1024 DATA 62A8A5D4D0379820
1025 DATA 8963C9FFF0B8A8A5
1026 DATA D3D016980A0A0A0A
1027 DATA 85CBA4CDB92D6529
1028 DATA 0F85CB992D654CF8
1029 DATA 6298290F85CBA4CD
1030 DATA B92D6529F005CB99
1031 DATA 2D654CF86298A4CD
1032 DATA 992D654CF862A200
1033 DATA A9FF85D5A9008555
1034 DATA A9168554A99E9D44
1035 DATA 03A9649D450320F3
1036 DATA 63A9008555A91685
1037 DATA 54A91F201564A9AD
1038 DATA 9D4403A9649D4503
1039 DATA A9009D4803A9009D
1040 DATA 490320EA63ADAD64
1041 DATA C92DF028C92DF030
1042 DATA C952F03CC957F03E
1043 DATA C944F040C953F04B
1044 DATA C951D00DA9028552
1045 DATA A21020E163202E64
1046 DATA 604C8660E6CED002
1047 DATA E6CF20BA624C8660
1048 DATA C6CEA5CEC9FFD002
1049 DATA C6CF20BA624C8660
1050 DATA 20BA624C8660203D
1051 DATA 644C866020BE61F0
1052 DATA 07A5CC85D020BA62
1053 DATA 4C866020BE61F00B
1054 DATA A5CC85CEA5CB85CF
1055 DATA 20BA624C8660A001
1056 DATA A20086CB86CCB9AD
1057 DATA 64C99DF004C84CC6
1058 DATA 61B8F02CB9AD6420
1059 DATA 8963C9FFF022488A
1060 DATA 2901F006680A0A0A
1061 DATA 0A488A4AD0086085
1062 DATA CC85CC4CFB616005
1063 DATA CB85CBE8E004D0D1
1064 DATA 8A60A5D4D024A5D3
1065 DATA 490185D3D0034C86
1066 DATA 60C6CDC6D130034C
1067 DATA 8660E6D4A90785D1
1068 DATA A5CD18690085CD4C
1069 DATA 8660C6CDC6D13003
1070 DATA 4C8660C6D4A90185
1071 DATA D34C1C62A5D4D028
1072 DATA A5D3490185D3F003
1073 DATA 4C8660E6CDE6D1A5
1074 DATA D1C90BF0034C8660
1075 DATA E6D4A90085D1A5CD
1076 DATA 38E90085CD4C8660
1077 DATA E6CDE6D1A5D1C908
1078 DATA F0034C8660C6D4A9
1079 DATA 0085D34C5A62A5CD
1080 DATA 38E90085CDC6D230
1081 DATA 034C8660A90F85D2
1082 DATA A5CD18698085CD4C
1083 DATA 8660A5CD18690885
1084 DATA CDE6D2A5D2C910F0
1085 DATA 034C8660A90085D2
1086 DATA A5CD38E90085CD4C
1087 DATA 8660203464202864
1088 DATA A9018554A9078555
1089 DATA A5D020B063A91285
1090 DATA 55A5CF20B063A5CE
1091 DATA 20B063A9FF8DFE02
1092 DATA 203F63A9008DFE02
1093 DATA 202E64A90085CD85
1094 DATA D185D285D385D460
1095 DATA 202864A9FF8DFE02
1096 DATA 205263A9008DFE02
1097 DATA 202E644C3C62A5D2
1098 DATA 1869058554A5D4D0
1099 DATA 12A5D185CB1865CB
1100 DATA 65CB690365D38555
1101 DATA 4C3263A5D118691B
1102 DATA 8555E655202E64A9
1103 DATA 1EA20020156460A9
1104 DATA 0085CD205263A5CD
1105 DATA 18690185CDC980D0
1106 DATA F260A5CD29F04A4A
1107 DATA 4A1869058554A5CD
1108 DATA 290785CB1865CB65
1109 DATA CB69038555A4CDB9
1110 DATA 2D6520B063A5CD29
1111 DATA 071869188555A4CD
1112 DATA B92D65A200201564
1113 DATA 60C9309020C93AB0
1114 DATA 0438E93060C94190
1115 DATA 14C947B00438E937
1116 DATA 60C9619008C967B0
1117 DATA 0438E95760A9FF60
1118 DATA 484A4A4A4A20BF63
1119 DATA 68290F20BF6360C9
1120 DATA 0AB009186930A200
1121 DATA 20156460186937A2
1122 DATA 002015646020E163
1123 DATA A9039D42032056E4
1124 DATA 60A90C9D42032056
1125 DATA E460A9059D420320
1126 DATA 56E460A9099D4203
1127 DATA A9FF9D48039D4903
1128 DATA 2056E460A9079D42
1129 DATA 03A9009D48039D49
1130 DATA 032056E46048A90B
1131 DATA 9D4203A9009D4803
1132 DATA 9D4903682056E460
1133 DATA A9FF8DF00260A900
1134 DATA 8DF00260A9522046
1135 DATA 642053E460A95720
1136 DATA 46642053E4608D02
1137 DATA 03A5D08D0103A92D
1138 DATA 8D0403A9658D0503
1139 DATA A5CE8D0A03A5CF8D
1140 DATA 0B03604472697665
1141 DATA 3A20303120536563
1142 DATA 746F723A20303030
1143 DATA 319B2020202B3020
1144 DATA 2B31202B32202B33
1145 DATA 202B34202B35202B
1146 DATA 36202B3720303132
1147 DATA 33343536379B436F
1148 DATA 6D6D616E643F9B45
1149 DATA 3A9B4B3A9B
```

BOOK REVIEW

By Matthew Preston

As a member of TWAUG you will know that the book library has just been established (for full details contact TWAUG) and I erge you to seriously consider joining as it is a real goldmine of information covering virtually every concept of the ATARI 8-bit, as I write this David has told me that another twenty books could be added to the collection of 60+ available, and so to the reviews;

The book being reviewed here is 'Best of PCW, SOFTWARE FOR THE ATARI XL'.

TITLE: "Best of PCW, SOFTWARE FOR THE ATARI XL"

AUTHOR: Various.
HIRE
PRICE: £1.50

This book is a real treat for users of BASIC it contains not only top quality type-in games but several utilities including a character sprite editor and a whole chapter full of hints and tips.

There are 20 chapters altogether, starting with typing tips, a list of how to obtain ALL the control characters and how to use the keyboard to the full. The others are as follows;

'BALLOON' a simple but addictive game guiding a balloon through caverns with a joystick.

'MUSIC MAKER' a utility to create simple tunes using BASIC.

'CALENDARS' a utility to print (yes I know it's obvious) calendars.

BOOK REVIEW continued

'MAP QUIZ' Tests your geographical knowledge of the British Isles. Harder than you think.

'ROBOTANK' a game using LOGO style commands to destroy targets and avoid obstacles on the screen.

'PATIENCE' a good version of the popular card game. Simple graphics but highly playable.

'VIDEO CREDITS GENERATOR' allows you to enter screens of text displayed in colour and then either scroll it or have it displayed screen, by screen. The results allow you to record your own credits on video tape.

'AIR SEA RESCUE' a simple game where you use a helicopter to pick-up people from one side of the screen and deposit them onto your lighthouse safe from the rocks.

'CHARACTER AND SPRITE EDITOR' a simple but quite useful editor for generating data statements required for BASIC programs. With this program you have the option of animating objects in memory so you can see if your animation is working.

'STOCKMARKET' a basic representation of a fictitious stockmarket allowing you to gamble safely knowing that you won't lose any real money.

'HANDEL' a program which plays a rendition of Handels' Arrival of the Queen of Sheba in BASIC.

'OMNIOPOLY' a version of the ever popular game MONOPOLY. The computer acts as banker and scorekeeper while you and a friend play against each other.

'SPACE MATHS' in this educational game you have to answer a number of mathematical problems in order to save your world from alien invasion.

'NO TRONS' a simple blast everthing in sight game.

'GO-MOKU' a version of the old oriental game GO-MOKU. The object of the game is to place your counters on a 19 by 19 board until you have a row of five not as easy as it sounds.

'ALIEN BLASTER' a version of space invaders with a twist, there are phantom invaders about son you don't know if you are hitting the real thing or not, just a bit tricky!

'PSEUDO DOS' obviously from the days before DOS 2.5 and ramdisks or TURBO BASIC, this neat utility is entered in from disk at line 30000. That way it should not obstruct your BASIC program in any way and you will have basic DOS commands available without having to go to DOS at all.

'COLOUR SELECTOR' a small program that enables you to select screen colours that you want and then be shown the poke codes that create them.

'ANIMATION' this is a brilliant piece of software and well worth the cost of the book itself. In short what it does is remove all

BOOK REVIEW continued

the donkey-work associated with animating Player Missile Graphics. Give the program information on size, colour, Player shapes and the software will seamlessly animate and move it around without slowing-down BASIC programs.

'AIRCRAFT LANDER' this is a simple flight simulation. What you have to do is land an areoplane safely, not at all easy in howling winds and a short runway, naff brakes and slow reaction times. A very satisfying game when you eventually manage to land with-out crashing.

'DUAL DISPLAY UTILITY' this utility is very clever in that it splits the refresh rate between two screens, but unfortunately I found the screen flicker unbearable.

'ROCKET ATTACK' this is a text game in then genre of the film 'WARGAMES'. You have to break into the Ministry of Defence computer and launch an attack on an alien invasion. I haven't got a copy of 'S.A.M.' (The Software Automatic Mouth) so I don't know if this part of the program works.

The hints and tips chapter is chock-full of excellent routines in BASIC that will save the novice programmer from having to do a lot of hard work, e.g. routines for clearing large amounts of memory, rapid relocation of the character set, auto-chaining of CSAVED programs, timer routines and an excellent PMG control routine. Further information includes a self-altering BASIC program technique and how to set-up VBI's.

In conclusion this book is most useful for novice and those experienced in programming. There is a wide variety of software included in the book most of good quality although you can't expect the games to be up to commercial standards nowadays because it was published in 1984. I still find the book useful, particularly for using programming ideas for my own use and if you don't feel up to typing in the listings, contact TWAUG because I think they have some of the programs on disk.

GAME REVIEWS

By Kevin Cooke

Yes, I'm back again for another batch of games reviews. Again, I've included a game for most tastes and pockets so that most people will be able to find something that they think sounds interesting enough to purchase. OK, here we go!

Title: ROSE GARDENS
Sold by: John Foskett,
26 Auckland Road,
Kingston-Upon-Thames,
Surrey KT1 3BQ,
ENGLAND.
Price: £1.00

Give personalisation name when ordering (minimum 3 letters, maximum 8 letters)

Well, I must admit that I was a little apprehensive about playing a text adventure. For those that don't know, adventure games usually contain no graphics but

GAME REVIEWS continued

simply text describing where the player currently is. The player could be in a dungeon, on a spaceship, in a sewer or.... in a rose garden! The object of the game is for the player to interact with the other people objects which he-she may find in an attempt to solve puzzles along the way, ultimately completing the quest which was set at the beginning. All actions are made by the player typing in what they wish to do, eg. "GO NORTH", "GET ROPE", "REPLACE THE GOLDEN SKULL WITH THE SAND BAG", etc. Usually the commands must consist of only two words, a verb and a noun, but sometimes the adventure will allow you to type in whole sentences. Generally I'm useless at the things and the only ones which I've managed to complete are a couple of Public Domain adventure games. However, I'm certainly glad that I gave this one a chance!

The game boots up with a very well designed title screen. Much effort must have been put into this alone as it is very impressive, especially considering it was created almost intirely out of the Atari's built-in CONTROL characters. The title screen isn't just a title screen but also doubles as a menu showing the available options. By pressing OPTION you can format a disk in a special format used by ROSE GARDENS to save games to later in the game. The other options are RUN DECODER and RUN ADVENTURE. I will return to the RUN DECODER option later on but for now will briefly describe the actual adventure.

When the adventure is first loaded in, you are presented with a very colourful screen and a message telling you to press start. Upon doing so, the details of the first location appears. In the top section of the screen is the location's description. In the section below this is a list of all exits from that location. The next two sections show any items of interest which the player may be able to interact with and the list of items which the player is currently holding (or wearing). The next section is empty apart from the "What now?" prompt, awaiting the player's command and the final five sections show the player's last 5 commands. As I mentioned above, each of these sections are in a different colour, making it very easy to differentiate between them. Also, it is very helpful having all of the room information on screen at all times as it saves using the LOOK command every couple of moves to take another look at what is in the room.

As the name suggests, the game starts off in a very large rose garden with you wandering around not having a clue what your task is. I won't go much further into what happens later into the game (I don't want to spoil it for you), needless to say that the player is NOT confined to the gardens for the entire game. Just use a bit of patience and you will soon find that the game livens up and becomes a LOT more exciting. John has obviously put a lot of effort into the storyline as it's a real surprise when.... no, I refuse to say!

GAME REVIEWS continued

The parser's understanding of the commands which you are likely to enter is good - if it doesn't understand what you are trying to say and you've tried re-phrasing it then you're probably trying to get it to do something which is not important.

While I think of it, I must go back to the RUN DECODER option on the menu. With ROSE GARDENS you are supplied with some help in case you get stuck in the game and can't progress. To prevent you from reading this accidentally (on purpose?!), it has been printed in code. To decode it, you simply type the series of letters into the decoder and it will print to the screen what the message is supposed to be, one letter at a time with each press of the start key. This gives you the option of seeing part or all of the clue. Again, a lot of thought must have gone into this option.

I understand that John is selling this as an example of his other text adventure games. At only 1 pound it's cheaper than most PD disks and is certainly superior in quality to most! By the way, if you supply John with a name (minimum of 3 letters, maximum of 8 letters) when ordering, then the adventure will communicate with you by name, eg. "Sorry, Ralph, but you can't do that!". I must admit that I really enjoyed this one, despite the fact that I'm usually useless at adventures. Yes, I haven't enjoyed myself so much for ages!

Title: ROBOMASH
Sold at: TWAUG
P.O Box 8
Wallsend
Tyne & Wear
NE28 6DQ.
Price: £2.50p

ROBOMASH is a new shareware game, released by TWAUG after it was sent to them by the author, Scott Johnson.

The story explaining the game concerns a nuclear power plant where an accident has taken place. A human team was sent to repair the damage but were killed by the security robots who have all had their circuits damaged by the radiation. The only man who survived the robots was fried in 10 seconds by the radiation! So in desperation, HOWIE, an experimental recovery droid has been sent in. Unfortunately, he is not fitted with weapons but must instead avoid the robots whilst collecting the plutonium and uranium. Cadium blocks can be pushed into the robot's paths, making them collide with each other and so die.

When you first boot up the disk, you are confronted with a menu which (very) slowly appears. Personally I would have liked this to appear a lot faster as I always want to see the options available before I pick one. However, you do soon learn them, I suppose, so it's not too much trouble.

When you choose to load the game, you are eventually greeted with the title screen showing

GAME REVIEWS continued

your character and the security robots which feature in the game. By pressing the SELECT key you can increase the game difficulty level (speed) but by pressing START you can go straight into the game.

The in-game graphics are not bad. All sprites are solid and move around the screen very smoothly without any flickering. Joystick response is also good meaning that you only die through stupidity rather than bad computer response. The main game is divided into two sections. The majority of the screen is taken up by the play area whilst the lower section advises the player of the time and lives remaining, no. of uranium-plutonium samples to collect, no. of security robots still to kill, etc.

There isn't really much more to say about this game. Graphically, the game isn't up to commercial game standards but in gameplay the game falls into a league of it's own! The number of times I had a "...last go" was incredible! It's also very difficult (I can't get past the first screen yet!) so it'll keep you occupied for ages. The addition of a level construction set will also increase the game's lifespan I think - maybe I'll be able to design a level which I can actually complete?!! Why not give it a go?

Title: TUBE BADDIES
Sold at: Richard GOre,
79 Sprotbrough Road,
Sprotbrough,
Doncaster DN5 8BW,
ENGLAND.

Another new game released by Richard Gorel I won't go into the long story of how he came about it, needless to say that I'm glad he did!

The amusing story concerns the Tubular Underworld and the fact that it is soon to be inspected by the "Big Inspector". Unfortunately, the tubes have been invaded by a number of Tube Baddies who have damaged the pipework. As Bip (and Bop also in a two player game) it is the player's job to kick the baddies out of the pipe system.

The game first loads up a lovely colourful title screen able to boast very large lettering and a smooth scrolling message at the bottom of the screen. When the fire button is pressed,, the main game screen is displayed. I must admit that when I first pressed fire I was very impressed by the graphics - a lot of work must have gone into them to make them of such a high cartoony standard. An excellent use has also been made of the available colours. Pipes are trailed around the main gameplay area, most containing holes from which the tube baddies emerge. By moving your character (which, again, is an excellently defined cartoony sprite) over the holes and pressing fire, large cartoony stick plasters are placed over them - I love it! By sealing off the holes in this way, it stops more baddies from emerging from them. At the bottom of the screen, are 8 pieces of pipe and a large cartoony bucket called Barney - he has eyes and a mouth!

GAME REVIEWS continued

To get rid of the baddies, the player must attempt to shoot them into either the pipes at the bottom of the screen. Extra points are gained if they are shot into Barney instead providing the level is completed before a timer runs out. Various usefull houses appear in a random place on the screen from time-to-time, often helping the players with their task. In between every couple of levels a VERY amusing screen appears. First your character drives across the top of the screen in a car, followed by the baddies! Again, this is all in a cartoony style and is great to watch. Your score and any bonuses gained are also displayed at the same time.

Overall, TUBE BADDIES is an excellent game which I would find very difficult to fault in any area. The graphics are excellent and the music is also very nice. The simultaneous two-player mode is also a great idea and really improves the game's addictiveness level. What can I say except this is a classic game if ever there was one!

WLOCZYKIJ REVIEWED

By Fred Meijer

After a very long time there's finally a new game and it's sold in Holland by Stichting Pokey (Pokey 'foundation'). This game was made in 1994 and is called Wloczykij. This appears to be pronounced as 'Wlooczycy' and the English translation is 'vagabond'.

WLOCZYKIJ REVIEWED continued

wlocz.pcx

The story in short: Wloczykij comes home after a big adventure. He only wants one thing: rest. So he decides to rest in his big lazy chair. Then the doorbell rings. At the front door is a strange looking being, it is an alien. His space-ship has crashed near the Earth. Fortunately none of his passengers were hurt. Unfortunately all the passengers are scattered across the Earth and they have to be found, because the people of the Earth are known for their fear for unknown beings. The alien asks Wloczykij for help. He does not have to think very long, he will help his alien friend.

You are controlling Wloczykij with the joystick. The game consists of various worlds. In every world you have to collect all alien travellers. This is accomplished by touching them. You can seperate the aliens from the (often very odd) people, because they are standing still and flashing.

WLOCZYKIJ REVIEWED continued

In every world you have to run and jump very carefully. Wloczykij jumps with somersaults, that makes it a bit difficult. Also in every world Wloczykij has to look out for witches, indians, plummers etc. Besides these people, he also has to watch out for fire and evil 'pacmans'.

When I saw this game for the first time in Germany, I was astounded. How was it possible that this game could have such enormous good graphics? Wow, what very, very good graphics. All figures are greatly detailed and an impossible amount of colours are used. You can see cleary what every figure represents. When I saw (and played) the game, I thought I was playing with a Nintendo.

The music is very good. It is written by Tomasz Liebich, who has written a lot more Atari 8-bit music, like Dark Abyss and Deimos. While playing the game you can choose between music or sound effects.

The packaging consists of the well-known Polish full-colour envelope. Specially for Stichting Pokey a Dutch manual has been made.

Title :Wloczykij
Software-house :L.K. Avalon
Price :Dfl. 15,- (ex. p.&p.)
Graphics :10
Sound : 8
Playabillity : 8
Value for money: 8
Total score : 9

NOTE: This article is a translation of the article which appeared in Pokey's Magazine. Wloczykij is not for sale yet in the U.K. Maybe Derek Fern will think about selling this game in the future?

CSS SUPER E-BURNER

by Barry Gordon

This article has been taken from Z*MAG online newsletter, issue 204.

A First Impression
February 1992

I was overwhelmed with excitement the day the UPS man delivered the box from Computer Software Services. And I had very good reason to be. After all, I had recently ordered one of their most technical devices, the "Super E-Burner". This device reads and programs EPROMS. That stands for (E)rasable (P)rogrammable (R)ead (O)nly (M)emory. Whoa! That's jumping in a bit too fast. How about we take a few steps back and run over some basic concepts first? Ok, here goes...

Inside your computer is a ROM (Read Only Memory) chip which holds the instructions that tell your computer what it is and how to operate. This Operating System chip (OS ROM) is a permanent component which does not need energy to maintain its data integrity. In other words, it doesn't go blank when the power is removed like a RAM chip. The OS ROM chip is programmed with

CSS SUPER E-BURNER continued

instructions when it is manufactured, and cannot be changed at a later time. An EPROM chip also retains its programming when the power is removed, however, its programming can be changed. An EPROM has a small window in the center of the top side of the chip. Erasing an EPROM is done by simply exposing this window to a high intensity UltraViolet light (Don't ask how, or we'll be here all day talking about transistor gate depletion levels and photo electron velocities). But programming (commonly called "burning") an EPROM chip requires a special device. Enter the Super E-Burner.

I had never seen a picture, nor heard a description of the physical aspects of the Super E-Burner. All I knew was it plugged into the cartridge port of the Atari 8-bit computer and was incredibly fast. After tearing the shipping box open, I pulled out and observed a rather odd looking device. The main unit of the Super E-Burner is a 5" x 5" PC board which holds 1 power connection, 4 IC chips, 48 miscellaneous electronic parts, 1 34-pin port, and 1 ZIF socket. What's a ZIF socket? Well, it stands for (Z)ero (I)nsertion (F)orce. It's an expensive socket that puts no pressure on the pins of chip. The socket has a small lever on its side. After placing a chip in the socket, pressing the lever down causes a metal plate in each hole to clamp down on the pins thus making a solid electrical contact. Coming off the 34-pin port is a two foot ribbon cable that connects to the cartridge interface board. This board resembles the insides of a disassembled game cartridge, and is used in a similar manner.

Despite the rather sparse and inadequate documentation which accompanies the Super E-Burner, operating the device is quite simple. The cartridge interface board is plugged into the cartridge port and the computer is booted with your favorite DOS. (For reasons unknown to myself nor Bob Puff, this device does not work properly with SpartaDOS X). Upon entering the cartridge, the following menu screen appears:

```
CO:01 NU:00 File:
Prom:0  Loc:00000  Write:00
Read:00
Type:     Retries:0000    Speed:1

   The SUPER-E BURNER 0.7
By: Robert Puff  (C) 1991 by CSS

 [A] 2732   25V
 [B] 2732A  21V
 [C] 2764   21V
 [D] 2764A  12V
 [E] 27128  21V
 [F] 27128A 12V
 [G] 27256  12V
 [H] 27512  12V
 [I] 27C101 12V
 [J] 27C301 12V (or mask ROM)

 Select PROM type >
```

From this menu, the EPROM size and programming voltage is entered. On this subject, the docs say nothing more than, "Select the proper EPROM type. An

CSS SUPER E-BURNER continued

incorrect selection of types can damage your EPROM." I recommend you find some other references to help determine the proper setting.
Once the EPROM type is chosen, the main menu is presented:

```
        Select Operation:
[R] Read EPROM      [B] Burn EPROM
[V] Verify EPROM    [E] Verify erase
[N] # of copies     [Q] quit to DOS
[S] Change speed    [ESC] New PROM
    size
[1-9] Disk directories

Please Select >
```

Although these menu selections are basically self explanatory, we'll run through them quickly:
[R]ead copies the information from a programmed EPROM onto a disk file.
[B]urn writes information from a disk file onto a blank EPROM.
[V]erify compares the information on a programmed EPROM to a disk file.
[E]rase checks to make sure an EPROM is blank.
[N]umber sets the number of EPROMS to be burned from a single disk file.
[Q]uit exits to DOS.
[S]peed alters the programming speed for older and slower EPROMS.
[ESC] goes back to the previous menu.
Four stapled pages accompany the Super E-Burner consisting of: A title page, a warranty page, a VERY oversimplified page of instructions, and a page showing the orientation of different sized chips in the ZIF socket. I immediately called CSS and asked Bob Puff, "Is this it!?" He assured me they would be revising the documentation soon (more on that later) and talked me through the necessary concepts and instructions. Now that I've had the Super E-Burner for about a month, I find myself rather comfortable with it. I have managed to backup every single ROM chip in the house onto my Hard Drive (that's over thirty chips including nine different Operating Systems for my 8-bit!) But the real fun of owning an EPROM burner is firm ware modifications. By disassembling, modifying, and rewriting parts of ROM code, I've managed to make my 130XE run in high speed mode with my US Doubler drives regardless of what disk-DOS I boot; I've made a stock 1050, US Doubler 1050, and an XF551 all respond to drive numbers higher than D4:; and I'm working on modifying my SpartaDOS X cartridge to use the standard SIO vector so it will work with a Multiplexer.
Within the documentation, a handwritten note says that CSS is working on a built-in editor which will support XE memory. This would make the Super E-Burner one of the most powerful firm ware devices I've seen. A ROM chip could be read, disassembled, modified, and rewritten without loading up any other programs. Bob said the revised documentation would be included with the enhanced editor.
All things considered, I am very impressed with the Super E-Burner. It is a well built, extremely fast, and (so far) reliable unit. In my opinion, it is well worth the price. (Which as of this writing is $169.95 + $8 S/H).

TWAUG ANNOUNCEMENT

PD SOFTWARE ON 3.5 DISKS.

Have you purchased the 3.5 drive interface from Micro Discount? If so, then you maybe interested in having software from our PD library put on to 3.5 disks. Any disk you order would be saved as Discom files so that on a 3.5 disk you would be able to get more PD software on one disk. The program DISCOM 3.2 would be included on all disks sent to you so that you will be able to unpack them at your leisure. If you are interested, just drop us a line and we will send you full details letting you know just how much it will cost per 3.5 disk and roughly how many PD disks you will be able to have on one 3.5 disk. This will of course depend upon the length of the programs you order but it will work out a lot cheaper than buying seperate disk titles. If you wish, we can also provide our newsletter disks on 3.5 disk for you, but as these disks are a bit more expensive than 5.25, there will be a small extra charge.

FOR PC XFORMER USERS.

If you have a PC and have purchased the PC Xformer, then you may be interested to know that we can now provide all our PD library titles on 3.5 disks saved as image files ready to use with PC Xformer. If you would like to know more, just drop us a line and we will send you full details.

THE TWAUG PD LIBRARY

Since TWAUG first released issue 1 newsletter and disk in January 1993, we have built up a PD library containing very nearly 500 titles. We are always on the search for new titles to add to the library, and hope to cover as many aspects of the Atari 8-bit as possible.

Although, the money we receive for subscription to TWAUG goes a long way towards the running cost, we do depend a lot on the sales of our PD disks to keep things running smoothly. We have always intended to keep the cost of subscribing to TWAUG as low as we possibly can, and we have no plans to increase the subscription in the near future unless it becomes absolutely necessary.

As you will have noticed, we are now increasing the size of the text in the newsletter which in turn means more pages. In time, we hope to be able to add a wider range of topics which will mean even more pages.

Although our list of subscribers grows with each new issue, we do not receive as many orders for PD software as we would like to see. Our thanks goes to all of those who have purchased PD disks from the library, your support is greatly appreciated. However, we would like to see more of you if possible, buying the odd one or two disks from the library and help us to work on more ideas that we have in providing more for our

TWAUG ANNOUNCEMENT continued

subscribers. We would like to be able to buy the right to some commercial software. At the moment, I am trying to track down some programs that have not been seen for some time now and I am sure many 8-bit users would be very interested in, but I would like to have money in the bank to be able to buy the copyrights.

Have a good look through all your back issues of the PD library and see if there is something that you feel you may be able to make use of. If you no longer have the updates, then just drop us a line and we'll send you a printed copy of the full PD library list.

THE TWAUG PRINT SHOP BOOKLET.

We believe that we have one of the largest collection of Print Shop icons, fonts and borders for the 8-bit. Due to recent additions to this catalogue, John is busy reprinting the booklet to bring it up to date. There will be quite a few extra pages in the new booklet, therefore, we will have to, unfortunately, increase the price. The Print Shop booklet will now cost £1.25, but it is worth every penny as it can not only help you decide which disks to buy, but it is also a very good quick reference guide when you are looking for one particular icon, font or border on the disks you have already purchased from the library.

THE ATARI 8-BIT BOOK LIBRARY.

In issue 15 of our newsletter, we announced that we were starting an Atari 8-bit book library. I am glad to say that it is now up and running, and we have a printed catalogue ready to send out to anyone who may be interested in having a look to see what books we have.

Many of the books in the library are very rare and cover a wide range of topics including programming in BASIC, Pascal, 'C', Logo and also Machine code. There are books to suit both beginners and advanced programmers. There are also books for those who just like to type in programs including some for children.

At the moment, we have over 60 books in the library, but more books are on the way from the U.S.A. Why not send for our catalogue, it costs nothing to have a look.

To become a member of the book library will cost £5.00, this is to give us money to buy even more books as and when they become available to us. Books can be hired for a period of one month for the cost of postage and packing which will depend on the weight of the book. It will range from £1.00 to £5.00. Some of the books we have are A4 size and contain over 400 pages. The hire charge for each book is given at the end of each book description in the library catalogue.

DISK CONTENT

Side A of this issue 17 disk has another good selection of programmes for you to enjoy.

This issue disk has been donated to TWAUG by Norman Williamson. The only exception is the 9 pin printer driver for the Citizen Swift Colour printer, it has been written by Ralph Bradley for TWAUG, short explanation for this driver is on the disk.

On side A is Calendar; if you've ever wondered on what day you were born why not find out with Calendar. You can go as far back as 1582 and to the year 2000. When the program runs all the instruction are displayed on screen.

Roll a dice; is a dice throwing game, just follow the prompts on the screen.

Fuelcheck; are you running a car? Why not use this program to keep a tab on the fuel consumption.

Multiformat is just what it says, you can use this formatting program to format as many disks as you wish without having to select the format option from the main DOS menu. It will format the disks in the density of the booted DOS.

Wallpapering; are you doing your own wallpapering? Then use this little program, by entering the measurements of the rooms, it will give you the amount of paper rolls you need.

Weddings; you've got a couple of options when you run this program, for instance there is a quiz for the anniversaries. Do you know what the anniversaries are called? Find out with this program.

Ludo chase; this is the game Ludo, to play it follow the prompts on the screen.

On side B of the disk there you find; NLIndex.Bas, this program gives you the titles you can find in the past issues of TWAUG.

GTIA256C, this is a demo of the 256 colours in your 8-bit.

Grapheze.Bas will only run when a printer is on line. I haven't tried it, but I believe it's a graphics program.

Tightrope; walk the tight rope and at the same time find out how fast your typing reaction is on the keyboard.

LACE

The London Atari Computer Enthusiasts.

As a member of LACE you receive a monthly newsletter and have access to a monthly meeting. They also support the ST with a good selection of PD sotware.

The membership is £8.00 annually.

Write to the secretary of LACE for more information:

Mr.Roger Lacey
LACE Secretary
41 Henryson Road
Crofton Park
London SE4 1HL
or phone: 0181-690 2548

OPERATING SYSTEM BLUES

By Mark Butler.

This article is reprinted from the AAAUA newsletter May/1995.

Suppose your operating system ran your car, What would a drive to the store be like? Here are some ideas.

MS-DOS: You get in the car and try to remember where you put the keys.

Microsoft Windows: You get in the car and drive to the store very slowly, because attached to the back of the car is a freight train.

Unix: You get in the car and type "grep st." After reaching speeds of 2000 miles per hours, you arrive at the barber shop.

Microsoft Windows NT: You get in the car and write a letter that says "go to the store." Then you get out and mail the letter to your dashboard.

IBM OS/2: After fuelling up with 6.000 gallons of gas, you get in the car and drive to the store with a motorcycle escort and a marching band in procession. Halfway there, the car blows up, killing everyone in town.

VM (mainframe): You get in the car and drive to the store. Halfway there you run out of gas. While walking the rest of the way, you are run over by kids on mopeds.

Amiga: You get in the car and begin driving very slowly. Halfway there the car crashes and is recalled, the salesman informs you that you need to buy a better model.

Atari ST: You walk out intending to go to the store, then you remember that there are no domestic cars, they are only sold in Europe.

Atari XL/XE: You get in the car and are dismayed to see a blank dashboard. But a dedicated user group member tells the car where to go.

As if that was not enough, how about these ones about how to program in different languages.

C: You shoot yourself in the foot.

Assembly: You crash the OS and overwrite the root disk. The system administrator arrives and shoots you in the foot. After a moments consideration of the work involved to get everyone back on-line, the administrator shoots himself in the foot.

C++: You accidently a dozen instances of yourself and shoot them all in the foot. Providing emergency medical care is impossible because you can't tell which are bitwise copies and which are just pointing at others and saying "that's me, over there."

Atari XE Basic: You seem unable to do anything but after writing several assembly routines coded as BASIC data statements and switching GTIA modes, you succeed in shooting yourself in the foot.

Ada: If you are dumb enough to actually use this language the US Department of Defence will kidnap you stand you up in front of a firing squad and tell the soldiers to shoot at your feet.

Modula/2: After realizing that you can't actually accomplish anything in this language, you shoot yourself in the head.
APL. You hear a gunshot and there's a whole in your foot, but you don't remember enough linear algebra to understand what happened.

Microsoft Visual Basic: You spend so much time playing with the graphics and windowing system that your boss shoots you in the foot, takes away your workstation and makes you develop in COBOL on a 3270 dumb terminal.
GFA Basic: After carefully assembling the gun and bullet from the building blocks provided, you shoot yourself in the foot faster than you could have imagined.

Atari ST Basic: You shoot at your foot, but the bullet travels so slowly that you have plenty of time to get out of the way.

Well, there they are... Hope you like them.

The Atari Classic Programmer's Club

This is to inform you membership is now only available on yearly subscriptions. The life membership option is no longer available. A member has access to various programmer services, a helpline, regular printed newsletter and discounts off our software?

Current UK membership cost is:

12 months: £6.00

Overseas members most welcome. If you require more details then please send an SAE to the address below. Overseas Atarians, please send two International Reply Coupons (available from your post office) for more information.

Still Available:

Swift Spreadsheet (Standard) - £9.95

Swift Spreadsheet (New) - £12.95

(NOTE: New version includes a revised 40 page A4 manual. For details of the standard version and of the program itself, please refer to the review in issue 67 of New Atari User).

Available Soon:

Menu Print: Still under development and currently undergoing its THIRD rewrite! We apologise for the delay but we expect the programming to be definitely completed by the end of September. More details will be published when ready.

THE ATARI CLASSIC PROGRAMMER'S CLUB
Pen-Tyddyn
Capel Coch
LLangefni
Anglesey, Gwynedd LL77 7UR
Wales