T — Y — N — E — & — W — E — A — R

TYNE & WEAR

ATARI 8-BIT USER GROUP

Newsletter of TWAUG

Software
Editorial
Buy & Sell
Hardware

TWAUG
8-BIT

Reviews
Help line
Section
Repair Info

Public Domain Library

ISSUE #4

JULY/AUGUST 1993

T — Y — N — E — & — W — E — A — R

U — S — E — R — G — R — O — U — P

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) 429-5639
FAX: (716) 247-7158
BBS: (716) 247-7157

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

TWAUG NEWSLETTER

EDITORIAL

Who to blame!!!

John Matthewson
David Ewens
Max Gerum

David would like to start a five liner section. Can you remember those five liners in the old Atari User Magazine? They used to be very interesting and I am sure a lot of you enjoyed them, so why not send us some of your creations for publication. Why not give it a try, the authors name will also be published.

COMPUTER SOFTWARE SERVICES Inc. from New York, famous for their BLACK BOX, gave T.W.A.U.G. permission to distribute their catalogue. Anyone interested in the catalogue please send a self addressed A4 envelope with a 28 pence stamp on to T.W.A.U.G., nothing more to pay.

Two members of the T.W.A.U.G. team have now got the BLACK BOX, that's John and Max. John has had his BB a while and it is fully operational, Max's isn't on line yet, the power unit must first be converted. That's the only drawback when you purchase electrical goods from the States. I hope to have part of the BB operational by the September issue. I will be writing about it in that issue.

T.W.A.U.G. can now also be contacted via COMPUSERVE on 100120,2025 just leave your messages.

The next issue will be ready by mid-September.

CONTENTS

TWAUG NEWSLETTER

Letter Section

To all our readers....

Hi there, this is Max calling for help. Is there nobody out there with any tips, comments or complains for our newsletter?

We haven't had a single letter for T.W.A.U.G. for this issue. David has had some letters but they weren't addressed to T.W.A.U.G., and I wouldn't publish any part of any letter that hasn't been approved by the writer.

Please tell us what you would like to see in the newsletter, or even let us know if you are satisfied with it. If you don't want your name published in full or not at all we will oblige.

I owe an apology to Paul for not publishing his letter in full in the last issue. The reason for that was if I had done so I would have had one or two blank pages. You will have noticed that our newsletter is on A4 size paper, and for each sheet we need enough material to fill four (4) pages. When we get near the end, before the publication date, there is usually no material left to put in the letter and it would be very difficult to fill the blank pages.

So I am appealing again to all T.W.A.U.G. readers please get your writing hand working and put pen to paper or keep tapping the keyboard.

BOOK REVIEW

A book is being written at the moment based on "Mapping the Atari" but it is solely for the XL/XE. It will be another month or so before we can give you more details about the book. It will be published and distributed by T.W.A.U.G..

I have been given some pages from the book to read through and to comment on it and I must say I am impressed by the details. The author has really gone out off his way to describe each memory location and explaining the functions with basic demo programs. With the few pages I received are a couple Appendice pages with very clear explanations and also included some programs in machine language. Anybody who is putting so much effort in writing a book for the XL/XE, to keep the 8-bit alive needs all the support we 8-bitters can give. I am not a programmer but I certainly will place my order even before this book is published.

If you have the book "Mapping the Atari" you will know that it contained some bugs, and it was mainly written for the old 800. Even the revised Edition has only a few pages included for the XL/XE. This new book will certainly be a saviour to the programmer with XL/XE's only, instead having to work out the locations or addresses, can just flik through the pages.

So why not get in touch with T.W.A.U.G. and let us know if you are interested in this new book. As soon as all the details have been worked out we will pass it on to you, probably in issue #5.

TEXTPRO WORD PROCESSOR

by Max Gerum

It is not very often that you read about a word processor and yet it is the most used utility in the computer world. In this issue I will be writing about TextPro, not much has ever been written about this word processor and yet it is easy to use.

There was a short article in Page 6 issue 57 by Edmund Blake, he said that it is 'the cheapest and best' word processor and I agree wholeheartedly with him, I use nothing else. I must have all the processors available for the 8-bit in my library and yet I prefer TextPro.

I had a number of requests for help with TextPro so I will try and be as clear as I can in explaining how to set up the configuration file and what TextPro can do in general.

TextPro is a binary load file, you will find, when you purchase the PD disk, the filename is TP.COM. You can either rename it to AUTORUN.SYS or run it with the 'L' option from Atari DOS or MyDOS, or just type TP and press Return when using SpartaDOS.

There are a number of TextPro versions about, the latest version is 5.0, I will be talking about version 4.54 and 4.56 here, there is a slight difference between them but the configuration is the same from version 4.5 onwards to version 5.0.

On the disk with the TextPro programme there are three other files, one is TEXTPRO.MAC, one TEXTPRO.FNT and one TEXTPRO.CNF. These three files load automatically when TP.COM is loaded.

TEXTPRO.MAC is the macro file, this adds special functions to the program.

TEXTPRO.FNT is any Graphics 0 font, this is used in the editor.

TEXTPRO.CNF is the configuration file, this file saves all of the user interface, printer and format selections. If these files are not on D1: which is the default drive, TextPro will load "bare" using the built in default values. You don't need to configure TextPro from the beginning as this can be done anytime. You can wait till you are more familiar with the functions of TP before you change the default configuration.

If you do want to configure TP I would suggest you just configure the printer margins and add a printer driver. Right, let's start by configuring the margins. Hold down the SELECT key during TEXTPRO startup to set the margins. The first prompt you will see is "Dec.# <l>" followed by the flashing cursor, now the default for the left margin is '5'. Just type your choice of left margin you prefer and press return. The cursor will move right and a "<y>" will be printed. This sets the 'z-margin', type in a new value and Return. If you wish to keep the default setting, which is '5' just press Return, and you will cycle through the other margins to set.

Here are the margins and default setting

<l> left margin 5
<y> z-margin 5
<r> right margin 75
<p> page length 66

TP WORD PROCESSOR continued

<t> top margin 5
<b> bottom margin 58
<s> line spacing 1
<x> line width 80

After you have set the last margin you will be returned to the editor. To save the new defaults for later use just save the config using the SELECT+CONTROL-S<ave config> command and type TEXTPRO.CNF and Return. You can use your new defaults right away without switching your computer off, simply press SELECT+CONTROL-L<oad config> command and type TEXTPRO.CNF plus Return, and the new default margins load into memory.

Why not setup a printer driver as well, you can either use a .MAC file or just type your code you wish to use into the editor. If you want to use the macro file you must make sure the MAKEPRT.MAC file is on the disk, then press CONTROL-V and type MAKEPRT.MAC and press Return, now press start and you will be presented with preset options, when you entered your last option place the cursor at the end of the line and press CTRL+W. Ignore the message in the command line, with this function your printer driver is now defined.

I setup my printer driver by typing the codes into the editor. The way I've done it is just as easy as using the macro file. Now what you do is type the character you wish to use in inverse and then use the equal sign and next the decimal number. Use the upper case characters for all your printer codes, the lower case characters have different functions. Here is an example I use 'U' for underlining, you would hold down the SELECT key and press 'U', this prints the 'U' in inverse now press the equal sign '=' and then type 45. This is how it will looks on the screen but don't forget the upper case U is in inverse  =45. Your printer manual will tell you that underline must be switched on and off, so therefore you must also set up a code for '1' on and a code for '0' off. After you've completed your driver you must place the cursor at the end of the line also and press CTRL-W.

You can carry on like that with all the printer commands you wish to have in your CNF file just in case you want to use them in the future. Now after you have entered all your printer codes again press SELECT+CONTROL-S to save the driver into the config file, just type: TEXTPRO.CNF and press return and it will be saved. If you want to continue using TP just load the saved config file back into memory by pressing SELECT+CONTROL-L(oad config) file and type TEXTPRO.CNF and press return.

You really don't need to read the entire documentation to begin using TextPro+. Just type your text into the editor, the words will wrap to the next line automatically, you only need to press return when you want to start a new paragraph. Let me give you the main commands, to load a saved file to the editor use the
CTRL-L<oad file> command.
When you press the CTRL-L the prompt "Load>D:" will appear on the top line. Just type the FILENAME.EXT<Return> of the file you want to load and the file will load to the editor. When loading a file to the editor the file loads from the cursor position down. Any text following the cursor will be erased, if the cursor is situated at the top of the file the entire text buffer is cleared during the load.

TP WORD PROCESSOR continued

To save a file press CTRL-S<ave file> command and the prompt "Save>D:" will appear on the command line, just type FILENAME.EXT<Return>. TP will save the file in the editor to disk. To get to the disk menu for loading or checking, just press CTRL-M<enu> command and the disk menu screen will appear.

At the top of the screen the disk content is displayed in three columns, a wide cursor bar is on the first filename. Move the cursor to the file you want, using the arrow keys without CTRL, to make a selection just highlight the file and press the character of the function chosen from the miniDOS at the bottom of the screen. You can "Load, Save, Delete, or View" any file from the directory you can also format a disk. To return to the editor press ESCape.

You will notice the bottom two lines of the directory displays the miniDOS system to use these functions press the characters that are highlighted. For Load, Delete and eXit use CONTROL and keys. If the DOS System your are using is recognised by TextPro all these DOS functions are supported.

Do you want to know on which line and page your cursor is when in the editor typing in a text? Press CTRL-W<here it's printing> and the command line displays 'page line and how many line there are to the page. For instance this line is line 40 of 58 on page 3 of this text.

CTRL-U<sed, unused memory> command and the size of the memory for the text buffer, the amount of memory used, and the position of the cursor in the file will appear on the command line.

CTRL-P<rint> command from the editor screen and the prompt "Print to>P:" will appear. The <P:>rinter is the default print device. To send the formatted text to the printer just press Return.

You can also print to the screen to get a look at the formatting by using the <E:>ditor as the print device. If you have a serial printer use the <R:>S232 device. You can also print to disk by using <D:FILENAME.EXT>.

If a prompt ends with ">" the selection requires a device, such as "D:". There will usually be a default device already provided but you can delete this and use another if you wish. A prompt that will appear often in TextPro is "Are you sure (Y/N)" Pressing the 'Y' key, either upper or lower case, selects the function. Any other keypress is a negative answer.

That's all there is to using TextPro+ in a hurry, but if you want to learn how to use all of the features of this powerful and useful programme, you must read the manual.

With the earlier versions of TP, up to 4.5, you couldn't save the printer driver into the CNF file, you had to load the driver into the editor and leave it there. Funnily enough it never took up much room, for about six lines of the printer driver only one line was registered.

There are more defaults that can be altered to suit your taste and fancy but I think I talk about them in the next issue.

You can use most DOS's with TextPro, if the DOS type is recognized support is provided for the full range of DOS features, including subdirectory support in both SpartaDOS and MyDOS. Binary load, verify

TP WORD PROCESSOR continued

toggle, and special directory listings are also supported. I would like to mention here that TP supports the RAMdisk if you have one. I use MyDOS 4.50 I find it very 'user friendly' this DOS supports the 1 Meg RAMdisk, and I also make use of subdirectories.

As you probably have gathered I do most of the printing for our newsletter and therefore I do like my large RAMdisk, to hold all the files and fonts when I do the printing. Let me tell you how to run TP, when starting up hold the Option key to disable BASIC, I don't need to depress option, I have no BASIC in the 1 Meg. So when I start, the DUP system comes on screen as soon as the RAMdisk is formatted, I then format a subdirectory first then I load all text files plus TP and Daisy Dot into the sub-dir with the 'C' option and all the fonts I use with DD3 I load into D8:, I use the wildcard mask it is much faster that way.

Working from the RAMdisk my two drives never run. When I want to use DD3, I use the macro file DD3.MAC, this file I load into memory as soon as TextPro is up and running, I load this macro file with CTRL-V(load macro) file. If I do want to use DD3 I just press OPTION+CTRL-P and DD3 comes up onto the screen immediately. You will find that Daisy-Dot is called PP.COM and this filename is used in the macro file DD3.MAC. So you must make sure your DD3 file, when using it with the macro file, is called PP.COM. Now coming out off DD3 I press 'X' to go to DOS and from there I use the 'L' option to run TextPro again, so you see I never switch the computer off when I move from one programme to the other and nothing is lost in the RAMdisk.

The manual of TextPro is explaining in great details all the functions and how to create your own macros. If you have received your TextPro copy from any User Group or down loaded it from BBS the documentation will be on the disk. Now before I finish the first part in this issue I will tell you what the manual consist off.

Table of Contents, it consists of 5 pages.
Section 1 TextPro+ Introduction: 7 pages.
Section 2 Editing: it has 9 pages.
Section 3 Keyboard Macros: 5 pages of explanation.
Section 4 Macro Commands: this has just over 7 pages of explanation.
Section 5 Additional Features 8 pages.
Section 6 TextPro+ Disk Menu: just over 6 pages.
Section 7 Printing: 13 pages on how to setup the print format commands.
Section 8 Configuring TextPro: this section has 7 pages.
Section 9 Special Configuration: again 7 pages.
Section 10 DOS Packages and TextPro+: 7 pages.
Section 11 Programmer's Notes: 6 pages.
Section 12 Command Summary: 4 pages of all commands available in TextPro, every body needs that section.
Appendix A: 6 pages explains about Bank switching, I will explain this part in next issue.
Appendix B: over 8 pages explains more features available including cross bank printing.
Appendix C: 8 pages on macro files available.
TextPro Macro System: 6 pages this macro system setup is available from our PD Library with the TextPro package.

This is all for this issue see you in September.

TWAUG NEWSLETTER

THE BASIC TUTORIAL PART 2

by Ofer Safemon

I'm back with another educational basic tutorial.

For anybody who felt that the first part was a stupid joke, well I decided to get serious.

First, we will discuss variables, but this time we will get very technical and show that variables are very useful stuff and can have surprising uses.

As a note for this time, and especially for the future: The programs following this article will concern the uses of strings in different ways. This does not mean that the programs will use only strings but other advanced programming, so it is important to understand the concepts I'll explain about strings and if necessary, the whole program. Everything will be understood at the end.

So lets get to work. Although I promised only variables today, I will discuss about a lot of tables that basic uses and not all for strings because the topics are close and usually go together.

Our ATARI Basic is very smart because it uses tokens. Every line, every letter, and every type of sign you put on the screen becomes tokenized by the computer in order to save space. Basicaly the computer exchanges the commands, and everything else we type, with predetermined codes and thus saves a lot of memory.

Altough I can add a lot of details about the tokenization process I think it is redundant since anybody who wants to build a detokenizer has to have a lot of knowledge in Basic and Assembler, and knowing about tokens wouldn't help you for much else.

Well, the story so far is very interesting (really?) but what about the variables I promised !!

Basic in its daily functions uses a lot of tables in order to be able to reference data in the best way. When we store a value in a variable, string, or array, Basic must be able to reference the variable and to store or retrieve the information. First it must be able to identify the type of variable. Then it must have memory set aside for it.

Each time we use a new variable in our program, we use 8 bytes of memory. A string and an array use the aditional size of the string or the array. The names of the variables are stored in a table. A second table stores the value of the variable or the location in memory that stores the string or array information. Because of the amount of memory that is used by variables, strings and arrays, it is recommended to re-use variable names whenever possible. For example if you need to use a timing loop 10 times in a program use a FOR-NEXT loop with the same variable.

However, it is better to use variables than numbers in a program if you will be using the number often because it saves memory, but that is another story.

The first table that Basic uses is THE VARIABLE NAME TABLE or VNTP in short. Every variable is assigned a number from 0 to 127. If you try to use more than 128 variables you will get an error message (don't panic because I still haven't encountered a program with even 100 variables, unless you are very wasteful).

Lets talk about addresses. The pointer to the beginning of the VNTP is 130,131 meaning PEEK(130)+256*PEEK(131) gives the start address for the VNTP. The variables in the table take a special form: simple variables appear as they are, string variables have the dollar sign '$' at their tail, and array variables have the paranthesis sign '(' at their tail. Also the last character of the variable is in inverse video to mark its end.

Once the variables have been stored in the VNTP Basic stores information about every variable in THE VARIABLE VALUE TABLE or VVTP in short. The variables are listed in the same order as in the VNTP. As I said earlier, every variable occupies 8 bytes of memory. Those 8 bytes are stored in the VVTP and now, we are going to check their meaning.

The first byte tells the computer what kind of variable it is: 0 for a numeric variable, 65 for arrays and 129 for strings (If you forgot to DIMension the variables yet used them anyway substract one from the above value. Those variables will not be usable but the computer inserts them anyway because they appeared and it should know about them).

The second byte is the position of the variable in the VNTP, or in english its location number (0 for the first variable, 1 for the second, 127 for the last, etc.)

If we are dealing with a numeric variable then the next 6 bytes are its value in BCD (Binary Coded Decimal). This deserves an explanation: The first byte is the exponent; 64 means 0,63 means -2,65 means +2, etc. Add 128 if the number is negative. The second gives the 2 decimal digits to the left of the decimal point; higher nible is the left most digit and lower nible is the folowing digit. The last 4 bytes give the 8 digits to the right of the decimal point. This seems very vague so it deserves some examples:

```
Number:0.02 = 2*10^-2
Format:63 02 00 00 00 00 (Decimal)
       40 02 00 00 00 00 (Hex)

Number:-0.02 = -2*10^-2
Format:191 02 00 00 00 00 (Decimal)
        BF 02 00 00 00 00 (Hex)

Number:460312 = 46.0312*10^4
Format:66 70 03 10 00 00 (Decimal)
       42 46 03 12 00 00 (Hex)
```

Back to the VVTP. If the variable isn't numerical, then the third and fourth bytes give an offset into the string/array table (which will be discussed later). If it's an array then the fifth and sixth bytes give the first dimension, and the seventh and eighth bytes give the second dimension.

Basic Tutorial continued

For example if we have an array A(3,4), the fifth and the seventh byte will be 0, the sixth byte will be 4, and the eighth byte will be a 5. This is because the array A(0,0) is a valid one so the addition of 1 is necessary in order to give the correct length.

If it is a string then the fifth and sixth bytes give its current length and the seventh and eighth byte give the dimensioned length.

The variable names and their values are all set (finally), now where's the program ?? in the STATEMENT TABLE or STMTAB in short. This is actually the table that holds the tokenized program we have discussed earlier.

This is pretty much the essence of my article. There are some more pointers which will not be covered because they really aren't so important.

To this point every sane person on earth that has read my article will ask: "SO WHAT ?", so I went out of my way to explain some very boring, dry and tedious stuff that nobody wants to know. What is it good for ? Well, here we come to the good stuff, there are some very cute programs on this disk that use exactly what I have explained.

First of all, as I promised, to all TURBO-BASIC users out there I have included a very nice program which has originally been taken from NEW ATARI USER. The program renames variables, meaning that if you have a program with variable names that you dislike you just load up the VARENAME and rename them. The program is relatively easy to use and should make no problems.

Another nice program is a program that uses a string to scroll messages on the screen. Although it isn't a smooth scroll, that wasn't the intention. The program changes the parameters of a string in VVTP to make the string believe that its string/array area is the screen.

The same principle is used in the third program, but this time its a demo of how to move player/missile graphics up/down without machine language. Don't worry if you haven't the faintest idea what is a player because one of the next articles will be about animation and I will discuss players and much more.

I have a few more things to add. First, for anybody who didn't know, the fastest way to fill a string with the same character in basic is:

```
10 DIM STRING$(X)
20 STRING$(1)="A":STRING$(X)="A"
30 STRING$(2)=STRING$
```

This will fill the string with the "A" character.

In TURBO-BASIC, if you need to fill the string with zeros ("") then you do this:

```
10 DIM STRING$(X)
20 STRING$(X)=""
```

That works because unlike Atari-Basic, TURBO-BASIC fills any dimensioned variable with zeros.

Altough I have discussed tokens and the STMTAB only in general I will give a very nice program to count the number of lines in your basic program:

```
30000 STMTAB = PEEK(136) + 256 * PEEK(137)
30010 LINES = 0
30020 NO = PEEK(STMTAB) + 256 * PEEK (STMTAB+1)
30030 IF NO = 30000 THEN ? "YOUR PROGRAM HAS "LINES;" LINES.":END
30040 STMTAB=STMTAB+PEEK(STMTAB+2)
30050 LINES=LINES+1
30060 GOTO 30020
```

This is a demonstration program and it works fine as it is but if you really want to benefit from it than put as many commands as posible on each line and renumber the lines so they will be as close as posible to 32767 which is the maximum line number in a program. Then change the IF statement in line 30030 to point to your starting line. You then LIST the program to disk, load your desired program and ENTER this one. Then do a GOTO to the starting line of your routine and it will count your lines.

I have another routine written in TURBO-BASIC that searches a certain string in a basic program, but it is packed with machine language, so if anybody is interested let me know and you can have it.

THE TURBO-BASIC TUTORIAL

I wasn't just going to leave all the TB freaks with only one TB program without a segment in this article.

Remember our topic-Variables, so lets discuss TB variables and get acquainted with a new beasty called PROCEDURES.

A procedure is in fact a subroutine like the ones we write and then GOSUB to them. Well here its a little different because every procedure goes like this:

```
PROC NAME
    .
    .
    .
    .
    .
    .
ENDPROC
```

Then when we want to execute them we just call:

```
EXEC NAME
```

Relatively simple, but as I am sure the next question that will be asked is, what's wrong with the old GOSUB-RETURN that we've used so far. The answer is: two things, first, order, and second, speed.

Basic Tutorial continued

Here I bring you back to variables. A procedure has a name which is entered into VNTP like any other variable. In the VVTP the first byte is the type of variable and is 193 for a procedure. The second byte as before is the number of variable. The third and fourth bytes point to the begining of the procedure in the statement table, but not as a line number, but as a memory location, and here is the huge advantage over the GOSUB statement, because if you do GOSUB 800, the basic will count lines in the statement table in a similiar way shown earlier, till it finds the line and if you have a very long program with subroutines at the end of it you lose valuable speed, but here the computer jumps directly to your procedure without waisting any time, no matter where your procedure is located.

Now, I have also said something about order. In TB if you type the following:

```
10 --
```

When you list the line, you will get:

```
10 ---------------------------------
```

So if you build your program nicely with procedures and divide them by lines, not only does the program look better, but EXEC LOAD has a meaning, while GOSUB 1000 is totaly vague.

Another useful thing in TB is a LINE LABEL. Consider the following:

```
GO #LABEL
    .
    .
    .
    .
    .
----------------
#LABEL
    .
    .
    .
    .
----------------
```

This is also very useful since first we have more order as explained earlier and the speed factor remains, while the normal GOTO command acts the same as GOSUB in terms of line searching.

In the VVTP the first byte is the type, and its number is 194. The second byte is the variable number. The third and fourth bytes are the location in the statement table were the label starts (memory location and not line number).

I think that will be about all for this time. I will conclude this long and a little difficult to understand article with a nice table that will sum up everything:

Byte	1	2	3	4	5	6	7	8
Numeric	0	V#	6 bytes BCD					
Array	65	v#	offset	dim1	dim2			
String	129	V#	offset	length	dim			
Procedure	193	V#	memloc	------	-----			
Label	194	V#	memloc	------	-----			

MEMORY LOCATIONS:

MEMORY LOCATION	POINTER
128,129	Basic Lomem
130,131	Start of VNTP
132,133	End of VNTP+1
134,135	Start of VVTP
136,137	Start of STMTAB
138,139	Start of CURSTM
140,141	String/Array table
142,143	Runtime Stack
144,145	Basic Memtop

NOTE.

There are three programmes on the disk to go with this article, they are SCRDEMO.BAS, PMDEMO.BAS and for Turbobasic VAREDIT.TUR.

Please take these three files of the T.W.A.U.G. disk to run.

TYNE&WEAR ATARI USER GROUP

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

YORKIE 256K plug-in memory upgrade.

Yes it's available once again, but in limited quantities.

The Yorky simply plugs into the PBI on the back of your 800XL (or internally upgraded to 64k, 600XL) to give you 256k of RAMBO XL compatible bank switched memory. NO soldering is required and you don't have to open up your machine.

It comes complete with a printed manual and a disk full of software designed to make use of the extra memory.

Price 50 pounds + 2 pounds p&p.

For enquiries and orders please write to:-

RICHARD GORE, 79 SPROTBROUGH ROAD, SPROTBROUGH, DONCASTER, DN5 8BW

or telephone me (Richard) on (0302) 784642 any weekend between 7pm Friday and 4pm Sunday.

NB. I can only guarantee this upgrade to work on computers that use a UK standard power supply.

FUTURA : XL/XE NEWSLETTER

Each issue is packed with

ARTICLES and SOFTWARE
Articles include 8-bit info, news, trivia, profiles, reviews, programming DIAMOND GOS and VCS columns, etc...
Software includes the latest UTILITIES, GAMES, and DEMOS.

Send now for the latest issue:
Disk.....£1.95
Printed copy & cassette.....£3.95
Cassette of programs.....£2.95

Please make cheques/POs payable to S.J.MURRAY
and send to -
Stuart J.Murray,
North of Scotland Atari User Group,
71 Walker Road,
TORRY, Aberdeen AB1 3DL

MICRO DISCOUNT

Now the largest Mail Order Stockist of Atari 8 Bit Hardware & Software in the U.K.

New Software for 1993

The last Guardian
Tagalon Disks £5.95 H & S Speed Tapes £4.95
Adax Disk £5.95
Eureka Disk £5.95
Guitar Wizard Disk £10.95
Hans Kloss Disk £5.95
and many more

Hardware & Upgrade Kits

Atari Disk drives (re-cond) £100.00
Data Recorders XC12 (re-cond)
Standard £28.00 Turbo £43.00
Upgrades from £18.95 to £51.95
and much, much more

To receive a 32 page catalogue you must first register your name and full postal address on our Data Base. The cost of this service is just £3.50 for 6 copies (1 every 8 weeks): £6.00 Europe: £10.00 Elsewhere. Prices on all items unless stated do not include post and packing, therefore please contact before ordering with details of your requirements to confirm availability and postage. Allow 28 days for delivery.

MICRO DISCOUNT
265 Chester Road
STREETLY
West Midlands B74 3EA
Tel:021-353-5730 or Fax:021-352-1669

NEW ATARI USER PAGE 6

The only magazine left in this country that supports the 8-bit

There is a large Public Domain Library available. Your support is needed to keep the magazine going.

It is now only available by subscription, for more details write to:

PAGE 6
P.O.BOX 54
STAFFORD ST16 1TB

INPUT - OUTPUT

on the ATARI using the Device Control Block

A device handler is a routine used by the operating system (OS) to control the transfer of data to and from a particular device for the task allotted. The devices supported by the Atari are:

S: (Screen)
P: (Printer)
E: (Screen Editor),
D: (Disk)
C: (Cassette)
R: (850 interface)
K: (Keyboard).

Memory addresses 768-831 decimal are used to hold vectors for the various handler routines, and this group is called the Device Control Block (DCB).

The OS disk handler routine is entered at address 58451, labelled DSKINV. By loading relevant values into DCB addresses 768-779, and exercising a jump to DSKINV from Basic, five disk operations can be executed. These are GET sector, PUT sector, PUT sector with verify, STATUS request and FORMAT entire disk. The five operations can be put to useful practical applications.

Not all the DCB addresses are user-alterable. Some are set automatically by the handler (OS), while others are unused. The DCB locations for disk operations, and the range of usable values are shown in Table 1.

The DCB can be set up to read a sector from the disk, and place the sector data into a reserved buffer area. Once the data is in memory it may be manipulated in a variety of ways. It can be displayed on the screen, the information can be changed and written back to the disk, it can be copied to a second disk, etc. The programs 'DCBDEMO1.DCB' and 'DCBDEMO2.DCB' allow sector contents to be displayed on the screen, the first in any order and the second in consecutive order from a chosen starting sector number.

On a DOS 2.0/2.5 formatted disk, sector 360 is the volume table of contents (VTOC), and sectors 361-368 form the directory of that disk. It might be interesting to scan file sectors to check the contents of a file. Directory data contains, in addition to the filename, information on the length of the file and its starting sector number. Use these programs to read the directory to file a file, then read the file itself. Sometimes apparent garbage is displayed, and sometimes intelligible words (such as clues to an adventure game) can be read. See references 1 and 2 for good discussions on disk (including directory) structure.

Table 1.1

DCB Locations - Disk Handler

Decimal	Label	Value	Explanation (dec)
768	DDEVIC	49-52	Drive 1 - 4
		64	Printer P1
		79	Printer P2
		80-83	RS232 Ports R1-R4
769	DUNIT	1-4	Drive 1-4
770	DCOMND	82	Read
		87	Write (verify)
		83	Status
		80	Putino verify)
		33	Format
771	DSTATS	?	Status code (code will be 1 if a successful sector read was executed)
772	DBUFLO	XXX	Lo/hi byte
773	DBUFHI	XXX	of the address of the source or destination of data to be moved e.g., disk sector data.
774	DTIMLO	31	Device timeout in one second units.
775	DUNUSE	---	Unused byte
776	DBYTLO	128	Lo/hi byte values
777	DBYTEHI	0	for number of bytes moved to or from data buffer (or disk).
778	DAUX1	XXX	Lo/hi byte
779	DAUX2	XXX	for sector number for read/write operation.

Table 1.2
DCB Locations - Disk Handler

Decimal	Hex	Label	User alterable
768	300	DDEVIC	No
769	301	DUNIT	Yes
770	302	DCOMND	Yes
771	303	DSTATS	No
772	304	DBUFLO	Yes
773	305	DBUFHI	Yes
774	306	DTIMLO	No
775	307	DUNUSE	
776	308	DBYTLO	No
777	309	DBYTEHI	
778	30A	DAUX1	Yes
779	30B	DAUX2	

INPUT - OUTPUT continued

One of the DCB operations is WRITE. DCBDEMO3.DCB is a sector copier. Up to 160 sectors per pass are read, their contents being stored in a long string in memory. The data is then written back to a blank formatted disk. Several passes are required, and the program runs more slowly than assembly code versions, but it gives a good idea of how these operations are carried out.

If we can copy sectors, it should be possible to alter them. DCBDEMO4.DCB is a one-at-a-time sector editor. Sector editors have many uses, including changing text in unlistable programs and customising displays. Antic's magazine disks contain an AUTORUN.SYS file which calls a menu program. This file contains the command RUN "D:MENU", and it is a simple matter to change the four letters M-E-N-U to your own four character filename - an effective autoboot technique for your own programs.

Finally, DCBDEMO5.DCB will print readable characters from any sector on a printer, an occasionally useful application of the Disk Handler DCB (such as printing hints and tips from some text adventure games!!)

Of course, using NOTE, POINT, GET byte and PUT byte commands from Basic much of the above operations could be carried out without resorting to use of the DCB. Programming in this case would be difficult and VERY slow. The DCB is a powerful technique in disk manipulation

In each of these listing, memory location 766 is poked with either a zero or a 1. Zero is the default value stored at this address. When 766 is poked with any non-zero value, those screen editor characters which normally must be preceeded by Escape (arrows, tab, clear screen etc), can be displayed. Thus the entire contents of a sector can be printed on the screen. The end-of-line character (character 155) is not overridden by changing location 766, and sectors containing character 155 (RETURN) may appear untidy when displayed.

WARNING!! Always use caution when writing to a disk as it is possible to destroy data under some circumstances. Even though these programs can be used to correct damage to a disk, it is advisable to experiment with back-up disks.

1. De Re Atari. Atari Program change, 1981.
2. Advanced Programming Techniques
   for your Atari. Linda Schreiber.
   Tab Books, 1983.
3. Mapping the Atari. Ian Chadwick.
   Compute! Books, 1983.
4. Machine LAnguage for Beginners.
   Richard Mansfield
   Compute! Books, 1983.

PROGRAMMING THE 6502

by Rodney Zaks (SYBEX Inc.)

A Book Review BY JEFF MADDOCKS

When learning to program in machine language it is essential to have a book or guide to learn from. In the Atari Assembler Editor Users' Manual it recommends three books, one of them being 'Programming the 6502' by Rodney Zaks. Zaks is well known in the computing field for his several books.

Programming the 6502' covers a broad range of 6502 topics. Zaks assumes that the reader has had no previous programming experience and begins by explaining several different number systems. (eg. Binary, Binary Coded Decimal, Hexadecimal), the basic computer system and how it interfaces with the 6502 chip.

He gradually introduces the early concepts, the instruction set, modes of addressing, a technical run-down on the chip itself, interfacing to other devices (for the hardware buff), data structures and program development.

Although this book is riddled with examples, diagrams and photos it isn't ATARI specific, so some of the examples will need to be modified. Pages often refer back to the one diagram.

A constant nuisance when you have to flip back several pages to find the correct diagram so that you can fully understand the explanation. To further that problem, when I first began Assembly Language programming myself, I found that the cryptic explanations needed several readings before I could grasp the idea.

Several of my friends have also ended up in the same confusion because of the same problem. I have also heard, from several sources, that the first edition of this book, as well as two subsequent editions have several errors in them.

A good book for beginners to start out with but more experienced users should look for some other book.

THE FIRST FIVE LINER

Our first five liner has been sent in by RANDALL FLAGG.
It's just a little GR. demo but at least it's a beginning.

```
10 REM CULUZ BY RANDALL FLAGG
20 GRAPHICS 0:POKE 559,0:FOR I=1536 TO 1549:READ A:POKE
I,A:NEXT I
30 DATA 72,173,11,212,141,10,212,10,10,141,24,208,104,64
40 DL=PEEK(560)+PEEK(561)*256:POKE DL+3,194:POKE 710,60:POKE
709,0
50 FOR I=DL+6 TO DL+28:POKE I,130:NEXT I:POKE 512,0:POKE
513,6:POKE 54286,192:POKE 559,34
```

ATARI CLASSIC PROGRAMMER'S CLUB

Mr. D. W. Davies.
Pen-Tyddyn
Capel Coch
Llangefni
Anglesey
Gwynedd
LL77 7UR.

Are you a programmer? or are you an Atari 8-BIT user who has often thought about wanting to write your own programme, but find you don't have enough knowledge to do so? Then why not get in touch with the ACPC and find out just what they can do to help.

The Atari Classic Programmer's club is a new group of 8-BIT enthusiasts who want to help build up the support for the Atari Classic Computer. They offer a wide range of support from giving advice on problems you may be having in putting your programme together, to helping you get your programme published when finished. No problem is too great, whether you are a good programmer or just a beginner, they will give you all the help they can.

Why not drop David Davies a line and ask him for his information sheet and order form, you won't be disappointed.

Membership charges are as follows:

UK & IRELAND

6 Months:	£3.50
12 Months:	£5.
Life:	£12.

EUROPE:

6 Months:	£5.
12 Months:	£9.
Life:	£15.

ELSEWHERE:

6 Months:	£6.50
12 Months:	£12.
Life:	£18.

There are a number of other services available including a programmer's helpline and a bi-monthly newsletter packed with tips on how to get the most out of your Atari. Why not get out the language of your choice, and join ACPC today.

NOTE from D. Ewens.

I have been in touch with David Davies both by letter and on the phone and I can say that he is a real genuine guy who, like TWAUG, is keen to do all he can to keep the 8-BIT alive. I hope that all of you who have had the idea now and then of having a go at writing your own programme, get in touch with him and see what he has to offer.

MARK'S GAMES COLUMN....

Alex of Moscow has once again provided an excellent map of a great Avalon/Micro Discount offering (many thanks Alex, your contribution is greatly appreciated) - Hans Kloss.

Talking of contributions, where are they? I have been waiting with baited breath, and waiting, and waiting. So come on gamers and adventurers - if we all do a bit we all gain a lot. Put pen to paper and help the guys at TWAUG keep the best newsletter available stay that way.

So, back to the matter in hand, Hans Kloss is a very reasonably priced game at £5.95 and is an absolute must for all keen gamers/adventurers. I have a great number of games, mostly adventures, and this one is the cream of arcade adventure.

You play the part of Hans Kloss, an under-cover agent for the allied forces during world war II. Rumour has it that the Nazis are building a powerful weapon in a secret bunker, to which you have gained access. It is your job to find a number of top secret documents to do with this project and then to destroy them.

Your adventure is fraught with danger negotiate roaming mechanised guards, and an abundance of traps such as trip operated machine guns (which always get the better of me). You will need to keep well fed and watered (this is usually the cause of my demised) and most importantly you must have a steady hand and nerves of steel. Anyway, enough talking, use the map and defeat the Nazis! Until Next time, good adventuring (well gaming in this case).

The map for this adventure is on the centre page.

THE OL'HACKERS
ATARI USER GROUP INC.

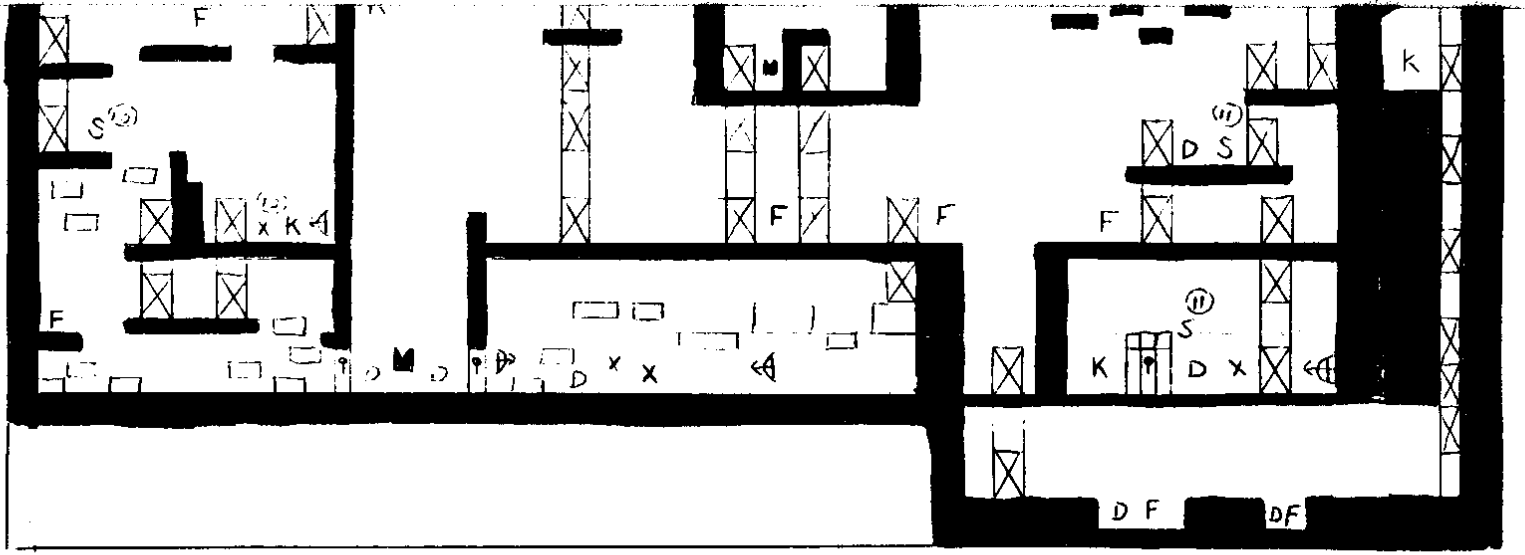
O.H.A.U.G. is an all 8-bit user group in the State of New York, they are producing a bi-monthly first class informative newsletter on disk.

The disk is double sided full of news, views, articles and bonus games and/or utilities. The disk has its own printing utility which you can use to read the content of the disk on screen or make hard copies.

A large PD Library is also available.

Some of the T.W.A.U.G. members are contributing to the OL'Hackers newsletter and the OL'Hackers are contributing to the T.W.A.U.G. missive.

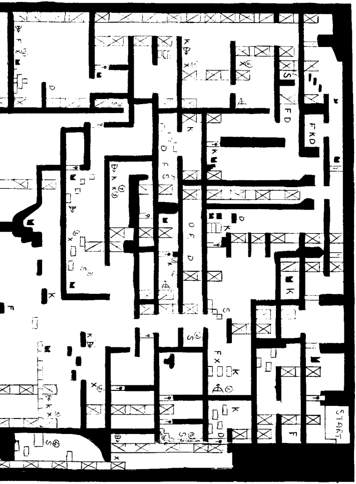
For more information on how to contribute to the newsletter write to:
O.H.A.U.G.
c/o A. Pignato
3376 Ocean Harbor Drive
Oceanside, N.Y.11572
U.S.A.



KEY

- = LOCKED DOOR.
- = ITEMS TO BE COLLECTED.
- F = FOOD
- D = DRINK
- X = TRAP
- S = ON/OFF SWITCH FOR TRAP.
- = GUN
- = LIFTS.

HANS KLOSS GAME MAP


START

KXP-1123 24pin Printer Driver.

PUBLIC DOMAIN by Richard GORE.

Exclusively for TWAUG.

The Panasonic KXP-1123 is a relatively inexpensive quality 24pin dot matrix printer. The text output is superb, all that is needed is a suitable Printer Driver. The one documented here is for use with AtariWriter+ and is in the Public Domain.

There are actually three versions each one having slightly different functions but the basic modes of operation are the same. They were created using the utility program supplied on the AW+ disk (accessed by holding down SELECT on bootup).

In all versions the standard options like underline, expanded and bold text are used exactly as in the normal operation of AW+, only the sub/superscript mode of operation is different. There are two ways of creating sub/superscrpits, one way is to use the special character set built into some printers (including the KXP-1123) and the other way is to forward or reverse half line feed the paper and print a normal character. This is the major difference between the printer drivers supplied in this package (details will be given later). You will probably be aware that AW+ has a function called "TYPE FONT', used by pressing CONTOL G followed by a number 1 to 9 to allow the selecting of various font styles and/or line widths (ie pica, elite). I have assigned these for each driver as detailed below.

The following text will give the individual details of the three drivers.

Filename: DRIVERA

Type Font	Function
1	PICA
2	ELITE
3	MICRON
4	COMPRESSED
5	Turn ITALIC on
6	COURIER font
7	PRESTIGE font
8	SCRIPT font
9	BOLD PS font

This driver employs the forward and reverse half line feed method for sub/super scripts. To select a super script press SELECT and UP ARROW (- key) and type your desired text, then press SELECT and DOWN ARROW (= key) when you have finished. To get subscripts use the keys the other way round, ie SELECT and DOWN ARROW first - your text - SELECT and UP ARROW.

There are two drawbacks to this method;

1) You can only use it with rear tractor paper feed and T.PUSH, top single sheet feed.

2) After the text has finished printing the printer will proceed to forward and reverse half line feed for a minute or so, I've no idea why but the way to stop it is take the printer 'off-line' and clear the buffer by pressing the FUNCTION and LF (Linefeed) together.

These problems are not major but the next method is better.

Filename: DRIVERC

Type Font	Function
1	PICA
2	ELITE
3	MICRON
4	P.SPACED
5	Turn SUB/SUPER scripts off.
6	COURIER font
7	PRESTIGE font
8	SCRIPT font
9	BOLD PS font

This version uses the inbuilt sub/super script fonts of the KXP-1123 printer. To use superscripts press SELECT and UP ARROW, type your text then press CONTROL G and 5 to turn them off. Subscrpits are done in a similar way but press SELECT and DOWN ARROW - text - CONTROL G then 5.

There are also two problems with this method;

1) Using above about 20 sub/super scripts will cause a lock up, which can be got out of using RESET. I've only come across this when using the XE version of AW+, maybe it's a bug in the XE version? It doesn't seem to happen in the XL version!

2) On starting a new page the printer selects the super script font and starts printing using that. Once again I've no idea why, but this can be overcome by using the print preview mode to see where the page boundaries are then putting CONTROL G and 5 before the text for the next page begins.

I believe this is the best mode for using sub/superscripts and it is the one I use for writing up my chemistry reports.

Filename: DRIVERX

Type Font	Function
1	PICA
2	ELITE
3	MICRON
4	COMPRESSED
5	P.SPACING
6	COURIER font
7	PRESTIGE font
8	SCRIPT font
9	BOLD PS font

This mode doesn't have any sub/superscript capability, so use this if you don't need them as none of the above mentioned problems occur!

Well thats about it, I hope you get some use out of these drivers. One final thing, although primarily intended for use with the Panasonic KXP-1123 printer it may work with others, especially the rest of the Panasonic range and most Epson compatibles like the Epson LQ850, indeed I would be glad to hear if it does work with other printers, as I am sure TWAUG will be.

If you want to contact me, feel free to write to me at:

Richard GORE, 79 SPROTBROUGH ROAD, SPROTBROUGH, DONCASTER, DN5 8BW

TOPSHELF instructions.

Topshelf is an easy-to-use, menu operated database programme with many high-powered features. Topshelf sorts your data in any way you like, saves your files on disk and retrieves them...even does arithmetic calculations. You can load Topshelf files into your word processor and format them into classy printouts, or you can print unformatted Topshelf reports direct from the programme.

USING TOPSHELF.

When you run Topshelf, it displays a white screen with a grey menu bar at the top. This is the first of two menu screens. Press [M] to see the second screen, press the [ESC] to return to the first.

First, get a sheet of paper and write a brief outline for your database. Draw four columns on your paper and label them FILENAME, DESC, SIZE and DISK #. These represent the four fields we'll be using.

Run the Topshelf programme. When the first menu bar appears, type [M] to access the second menu. Now press [C] to Create a new data file.

The Words NO, TITLE, LENGTH, TYPE (C/N/F/S) and DEC should now be displayed at the top of the screen. These will help you design each field in the database.

The TITLE of the first field on our paper database outline is FILENAME. Type FILENAME and press [RETURN]. The cursor will apear beneath the word LENGTH. Since Atari filenames cannot be longer then twelve characters, type a 12 in this column and then press [RETURN].

The cursor will appear beneath the letters TYPE [C/N/F/S]. This lets you define the type of information to be kept in the Field. [C] gives you Character text data and [N] stands for Numeric data. (Later in this article we'll explain the other choices, [F] for Function and [S] for Statement).

Since filenames are strings of characters, type [C] and [RETURN]. Since the DECimal column is only used with numeric data, the cursor returns to the TITLE column. the FILENAME field is complete.

The second field on our paper database is called DESC, it will be a one word-description of the file you are indexing--BASIC, text, picture,etc. The TITLE for this field will be DESC. We'll arbitrarily assign it a LENGTH value of 8. Since DESC is a character field, enter a [C] in the TYPE category.

The third field on our paper, ZIZE, is the size of the file in sectors. Type the word SIZE in the TITLE column, and press [RETURN]. Next, fill in the value of the LENGTH column. Since you're not likely to come across a file larger than 999 sectors, you'll only need three places to hold your SIZE values. Type 3 and press [RETURN].

Since the SIZE field will only hold numbers, it is a NUMERIC field. Type an [N] in the TITLE column and press [RETURN]. This time, the cursor moves to the DECimal column.

DEC lets you specify the number of decimal places you'll use in arithmatic calculations. For example, if you're dealing with pounds and pence you'd type 2. This would actually add three spaces to your SIZE value--two number spaces plus a third for the decimal point. Since our sector counts will always be integer, type [0] in the DEC column and press [RETURN].

Our final field will hold the disk identification number, titled disk #. Typing a 4 here will give you a range from 1 to 9999 disk. More than enough. As this is a numeric field, type [N] in the TYPE column, then type the number of decimal places in the DEC column.

We've created our four fields. At this point, the cursor is on the 5 of the TITLE column. Press [RETURN] to go back to the main menu.

ENTERING DATA.

Adding data to the database is as easy as completing a form. From the main menu, Type [A] to add data. Field by Field, the programme will prompt you to type the data. Using the above example, the programe would prompt you for a FILENAME, the DESC (the description), the SIZE and the DISK #. Remember to SAVE your data file regularly!

You can also use the Create option to change the fields of an existing database. For example, you could use the command the name of your SIZE field to COUNT or AMT. But be cautious; if you press the [SPACEBAR] twice while in the TITLE column, you will erase the entire field and every field below it!

Practice using the functions described below.

VIEW lets you look through data which has been stored on the field of your choice. When data is on display, another menu appears. Pressing the [SPACEBAR] shows either the next matching data item or the END OF DATA statement and the main menu. Previous data remains on the screen. To clear the screen, press the [ESC] key.

BROUSE shows your records one at a time--from either the beginning, middle or end of file.

ADD lets you put a new data item into your file.

SAVE will store your file on disk. But don't type the "D: before the filename, the programme does this automatically.

LOAD retrieves data files that have been previously saved to disk.

MORE displays the second part of the main menu.

To exit the programme, you could press the [RESET] key, but this way is so much more dignified.

SECOND MENU.

The second menu screen contains the following options:

TOPSHELF continued

SORT, which sorts your data on which ever field you select.

FIND, searches the database for matching data. For an exact match, you must use quotes at the defining and end of the search string. When the data has been found, another menu appears asking if you want to DISPLAY, PRINT or SAVE it.

The PRINT selection transfers unformatted data to virtually any printer. The SAVE choice lets you store the data on disk in either of two ways. You can save it as a standard ASCII text file--for use with most word processors, or you can save it as data and create a new data file.

You can display the DIRECTORY of the disk in any drive you select.

On several menu selections you'll see +:-:RETURN ENTERS. This means that if you press [\], the next field is displayed on the top of the screen and the bar moves down one. Pressing [-] moves the bar up one field. Pressing [RETURN] indicates that this is the field you want to use.

When you VIEW or BROUSE the file, you'll see a new menu whenever data appears. This is for corrections. You advance one at a time with + or - and using the [SPACEBAR] to continue within the field you selected. Pressing [ESC] from any blinking cursor clears the screen and returns you to the main menu. Pressing [CONTROL] [P] transfers the data on the screen to the printer.

FUNCTIONS, STATEMENTS

Sometimes the value of one field is based on the values of previous fields. A database which computes grade averages is a good example of this. In this case, the final grade is determined by a function (FORMULA) which uses grade values from previous fields.

The [F] key lets you do simple mathematical Functions on one or more fields. This only works on numbers within fields. And the data must have been entered before the maths function was selected. You programme the function by typing the field number and the operation to be performed. For example, 1+2x3-4 means, "take field 1, add it to field two, multiply that by field 3 and subtract field 4 from the total." You must start with a field number and end with a field number. To programme your function, you've got 20 bytes for each field.

Also, Topshelf Functions cannot use any constants--such as a 7% sales tax. But there is a way to get around this limitation. If you need to use a constant for your calculations, select the field type as Statement. This lets you enter anything you want in the field, be it a number or a comment. Your entry will be displayed automatically whenever you add data to the file. If it's a number, you can have a function calculate it just like any other data. Otherwise, the Statement will be treated as a comment.

WORD WEAVE instructions

Here's a specialized word processor for creating a branching text stream - be it interactive fiction or a host of practical multiple-path applications.

We all know the story:

Rapunzel was locked high in a tower by a wicked witch. Every morning the witch would go to the foot of the tower and cry:

Rapunzel, Rapunzel - let down your golden hair!

Rapunzel would let her long braids drop through the window, and the witch would climb up.

One day the witch demanded that Rapunzel let down her hair as usual. But when Rapunzel came to the window, the witch saw that she had gotten a Mohawk.

Hold it! A Mohawk?! Well, perhaps another cut would do - or an entirely different twist: Rapunzel dumps a pail of dirty water! And then what happens? Ahh, what a tangled plot we can weave - with WordWeave....

WOVEN TEXT:

WordWeave is a structured word processor for creating interactive stories and other multiple-path text applications. These stories have a variety of surprising turns and numerous endings, because they allow the reader to make decisions as the plot develops. Do you want Rapunzel to let down her hair, or would you prefer she got a Mohawk? From each page, the text branches to a new page - and a new turn of events - depending on the choice of action.

With WordWeave, all kinds of alternative paths are possible. Tracing through a woven fabric of text, both the writer and the reader have the freedom to determine how a "story" may develop, page by page. In this program, a page consists of one screen of text and the option to branch to four different pages.

To use WordWeave, load and run the program, and then remove the program disk from the drive - it is no longer required. WordWeave operates from one menu containing the following options:

Read
Write
Print
Prepare Disk
Quit

PREPARING THE LOOM:

Before you can create a story, you must prepare a WordWeave data disk using the menu's Prepare Disk option. The disk that you prepare should be a blank initialized disk. You will need one disk for each text file that you write.

When you choose the Prepare Disk option, the program prompts you for a title and an author. These two items are the first to appear on the screen whenever you choose to Read or Write with WordWeave.

WORD WEAVE continued

WRITE:

Once you have prepared a data disk, you are ready to start weaving your multiple-path text. Press RETURN to turn the page and start writing.

In both the Write and Read options, the screen always displays the page number in the upper-left corner of the screen. The upper-right corner shows the number of options that are available to the reader. Each time you, the writer, add another branch from this page, the number of options increments.

Write mode provides several editing options: You can enter a screen of text, define the different pages that this page may branch to, go back to the previous page, go to any desired page, or return to the menu. There is a different keypress for each of these options (See control capsule, below).

When writing a page of text, you have all the standard editing features (i.e., insert, delete, etc.). You can even add a header that titles each of the different pages.

A simple keypress allows you to toggle the Branches menu on and off. This menu appears at the bottom of the screen and displays the various Branch options that are available to the reader. You can specify up to four branches, each of which can branch to any page. You can even have a page branch back to itself. You do not have to use all four branches - in fact, you don't have to use any. Here is an example of a Branch menu:

```
----------BRANCHES----------
1- Have Rapunzel let down her hair
2- Give Rapunzel a Mohawk
3- Pick a new hairstyle
4- Dump the Water
```

To edit the Branches menu, press CTRL-B. To add a description to a branch, such as "Have Rapunzel let down her hair," simply press the key corresponding to the branch that you wish to edit and enter your description. To designate which page an option branches to, put the cursor on that Branch line (1-4) and use the "Branch Destination" command (CTRL-G). Now the program prompts you for the page number that this branch should go to. If you do not enter a number, the computer defaults to the next available empty page - or, if you have previously entered a number for this branch, it defaults to that page number. Once you enter the number, WordWeave immediately takes you to that page. If you wish to "prune" a branch from a page (that is, erase a branch previously set), simply enter a page number of zero when specifying where the branch goes to - effectively truncating that branch from the page.

If, for any one page, you have more than four branches in mind, use the fourth branch to go to another page, where you may continue to present alternatives.

If you want to go back to the previous page, press ESCAPE. To go to any page at all, make sure that the Branch menu is not present, press CTRL-G, enter the desired page number, and away you go. To return to the main menu, press CTRL-X.

Now it's time to start writing! Be creative and experiment. For example, try making your story into a maze. Pretend that the pages are rooms and each of the four branches are exits - north, south, east, and west. The possibilities go on and on.

READ:

Before you choose this option, make sure that the data disk with the text that you wish to read is in the drive. The first thing that you see on the screen when you choose the Read option is the title page. Press RETURN to turn the page.

In this mode, you can display the Branches menu, select a Branch option, go back to the previous page, or return to the main menu. There is a different keypress for each of these options.

From Read mode, you can venture forth into a story's every turn, twist, and pitfall. You control the events by making your own choices as to which direction the story branches. If you don't like the way the plot is going, you can always return to the previous page and make a different decision. If you don't like the way the story is reading, you can always go back into the Write mode and change it!

PRINT:

With this menu selection, you can print all or part of a WordWeave story. When choosing this option, make sure that your data disk is in the drive. Because the print routine is set up to print two pages of text per one sheet of paper, be sure to begin the printout at the top of the sheet. The computer asks you for the starting and ending numbers of the pages that you wish to print. Once you have entered this data, each page is printed, including the Branches menu, which specifies the page number that each option branches to.

QUIT:

Select this option to exit the program. Don't worry: You are in no danger of losing data when you exit the program. All of a story's information is kept on your data disk.

NOTES:

During the Prepare Disk option, the disk is initialized and all previous files on the disk are erased, so don't use a disk that has anything on it you want to keep. Only one WordWeave file can be stored on the disk, so don't use the disk for anything except the one WordWeave file. If you look at the directory of the disk with DOS, it appears to have nothing on it - always place a label on a WordWeave disk so you don't accidentally use it for something else.

WORDWEAVE continued

Due to this program's unique method of screen memory management, it works ONLY ON XL and XE models.

CONTROL CAPSULE:

Write Mode:

CTRL-H: Edit header
RETURN: Exit header
CTRL-G: Go to a page
ESCAPE: Go to previous page
CTRL-B: Toggle Branches menu
CTRL-DELETE: Delete character
CTRL-INSERT: Insert space
CTRL-X: Exit to main menu from any page except title page

Editing Branches Menu:

CTRL-B: Toggle Branches menu
RETURN: Edit Branches 1-4
CTRL-G: Define a branch destination
CTRL-X Exit to main menu

Read Mode:

CTRL-B: Toggle Branches menu
1 - 4: Choose Branches 1-4
ESCAPE: Go to previous page
CTRL-X: Exit to main menu from any page except title page

HINTS & TIPS.

by Dave Ewens.

First of all, I'd like to thank those of you who wrote to me saying how you enjoyed my little article in the last issue.

The idea for this column came to me when looking through some old mags, I remembered just how helpful some of the bits and pieces I'd picked up from them over the years, had been to me. Maybe you have some useful tips of your own that you'd like to pass on. If so, please send it in and I'll be only too pleased to include it in my column.

NOISY I/O FLAG.

If you would like to turn off the sound of data pulses coming out of your TV or monitor speaker during disk or cassette I/O, type POKE 65,0. Bring it back with POKE 65,1.

BYE BYE REBOOT.

Here's a tricky way to return an AUTORUN.SYS file without turning off your Atari XL or XE. Press (RESET), type in BYE and press (RETURN). You will see the SelfTest menu. Press (RESET) again and AUTORUN.SYS will take off.

HINTS & TIPS continued

BINARY LOCATING.

Want to find out where a binary programme is being loaded? The first six bytes will give you the answer. The following programme reads them.

```
10 OPEN #2,4,0,"D:YOURPROG.OBJ
20 FOR I=1 TO 6:GET #2,A
30 PRINT A:NEXT I:CLOSE #2
```

Ignore the first two values which will be 255. Multiply the fourth value by 256 and add the third byte to the result. You now have the starting address. Repeat the above for values five and six to find the ending address of the binary programme.

ML DATA LOADER.

DATAMAKR.BAS is a short programme which converts machine language object files into BASIC DATA statements that can be ENTERed and used in your own BASIC programmes.

Just insert the filename of your object code in line 20. Then choose a Filename for your DATA statements and type it in at line 15. DATAMAKR will read your object code from the first file, and write BASIC DATA statements to the second.

When run, DATAMAKR asks for a starting line number. DATAMAKR will use that line number to creat its first DATA statement. Subsequent DATA lines will be numbered in increments of ten. If you prefer to number your programme differently, change the 10 in line 110 to a value of your choosing.
After DATAMAKR has written your file of DATA statements, type NEW and ENTER it into RAM. Remember that you'll probably have to delete the first six items of data, which are file header bytes. You may also find a comma after the last entry. This should be deleted too.

```
10 DIM LINE$(127):TRAP 120
15 OPEN #3, 8,0,"DATALIST.LST"
20 OPEN #4,4,0,"D:FILENAME.OBJ"
30 ? CHR$(125);"STARTING LINE # = ";
40 INPUT LN
50 LINE$=STR$(LN)
55 LINE$(LEN(LINE$)+1)="DATA "
60 FOR J=1 TO 15
65 GET #4,D
70 LINE$(LEN(LINE$)+1)=STR$(D)
75 LINE$(LEN(LINE$)+1)=","
80 NEXT J:GET #4,D
90 LINE$(LEN(LINE$)+1)=STR$(D)
100 PRINT #3;LINE$;LINE$;" "
110 LN=LN+10:GOTO 50
120 PRINT #3;LINE$:CLOSE #3:CLOSE #4
```

BUY, SALE & CONTACT SECTION

WANTED

WANTED TO COMPLETE MY COLLECTION
Page 6 issues numbers 1, 2, 4, 5, 11, 12, 13, 14
reasonable price or swap for Antic or Analog magazines
Ring for details TOM:
on 091-5869425

Does anyone have or know how to get a BOOK called RANDOM ALLEY Adventures for the Atari 8-bit. If so please call:
Michael on 091- 2859356

FOR SALE

1020 PRINTER PLOTTER with pens & paper and power unit also demo Cassette, instructions - all boxed.
£25.00 including post & packing
Tel: Mark on 0602-720597

A selection of books for the Atari XL/XE computer. I also have quite a few original games, complete with instructions, a few without instructions. Send S.A.E for lists.
and
ATARI TOUCH TABLET complete with Atari Artist cartridge and instructions. Still in original box, £20. Light Gun and Bug Hunt cartridge, £15. Prices include postage and packing.
TEL. 091/2710086, or write to:
D. Ewens, 48 Fouracres Rd. Cowgate, Newcastle-On-Tyne. NE5 3AX.

ATARI RADIO USER GROUP.

Attention all you Radio Hams with Atari 8-BIT computers. Do you already have your Atari hooked up to your radio system or do you wish you could? In either case, why not join us. At the moment, we do not have many members with 8-BITs, but hopefully that will soon change. Maybe someone out there could let us know where we can obtain interface/software for running the Atari 8-BIT with the radio setup. Any information you can give would be very welcome so we can help those who would like to get hooked up but have been unable to so far. We will be producing a quarterly newsletter (if we get enough interest), so why not get in touch and let us know what equipment you have. Please contact:
Graham Rayner G7KCT, 38 Brockhurst Rd., Chesham, Bucks. HP5 3JE.

ERRATA.

One of our subscribers found an error in Nir Darey's article and this error can be found under section 3, File Sectors. This is what our reader is saying:

In Nir's article under 'file sectors', He claims that "byte 125 contains the total number of bytes that are actually used". This is wrong it is actually byte 127. It is also bytes 125 and 126 that make up the 10-bit number not bytes 126 and 127.

TWAUG NEWSLETTER

CRACKING THE CODE

by Keith Mayhew and Roy Smith

Re-printed by M.Gerum

This article first appeared in "The U.K. ATARI Computer Owners Club" later renamed "MONITOR".

PART 4

More Logical Operators

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

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

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

DECISION MAKING?

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

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

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

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

CRACKING THE CODE

continued from previous page

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

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

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

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

2. If the accumulator is equal to the data then the result will be zero and the carry will be set, i.e. Z=1 and C=1

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

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

Stack Operations

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

Interrupt Instructions

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

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

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

Bye for now!

CRACKING THE CODE

continued from previous page

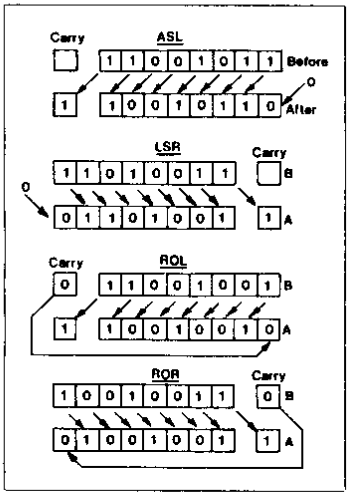


Figure 1.

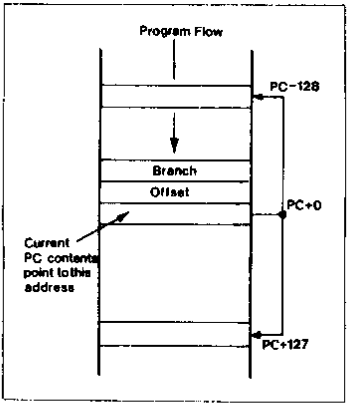


Figure 2.

TRANSFER OPERATIONS

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

ARITHMETIC OPERATIONS

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

LOGIC OPERATIONS

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

COMPARISON OPERATIONS

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

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

Table 1.

CRACKING THE CODE

continued from previous page

CONTROL OPERATIONS

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

BRANCH OPERATIONS

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

Table 1.

HEX TO DECIMAL CONVERSION

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

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

Table 2.

INTO THE EAGLES NEST

REVIEWED by Mark Fenwick.

Into The Eagles Nest from Atari Corp is probably the last or one of the last Roms produced by Atari for the 8-Bit.

Your mission is to enter the Eagles Nest, the castle the Nazis are using as a secret fortress. Allied saboteurs, before you, have planted bombs around the castle. You must rescue prisoners held there and destroy the castle with the bombs which have been hidden for you.

Once you've inserted your Into The Eagles Nest Rom you'll be confronted with the games credits and then the title screen. The title screen is well detailed with the castle in the background. You are then shown the high score table which is full of not bogus names as may have been thought but the word 'Pandora'. Pressing the fire button takes you to the option screen where you may select a mission 1-4. In missions 1-3 you must rescue prisoners held in the castle with greater degrees of difficulty. Mission 4 you must blow up the castle by setting off the various explosives around the castle on each floor.

Selecting mission 1 takes you straight to the gameplay. The far right of the screen shows your status ie. Keys collected, Score, Ammo and number of hits. These refer to the number of times you've been shot, should you be hit fifty times then the game will end. To the left of this is the playing area, Into The Eagles Nest is similar to Gauntlet in looks, as play is taken from an overhead prospective. The bottom of the screen has a small space for scrolling screen prompts for guidance during your mission.

You start off your mission in the Store room (where you secretly crept in). From here you can see parts of the other rooms to be explored, all divided by stone walls, floors can be seen as small cobbles. To the left of you is a key, while in front of you is a door with rather a lot of Nazis behind it. To pick up a key simply walk over it. You'll only need keys for opening grey steel doors, other yellow/orange wooden doors can be opened by shooting at them. Movement is controlled via the joystick with the button used for shooting as well as opening chests.

During your mission you must enter all rooms on each of the four floors. You must retrieve as many art treasures as possible as well as ammo when required. Your health level (Hits) will soon increase during your journey so to help you scateered around are first aid kits and cold foods. Touching these will heal a certain amount of hits. Also scattered around the castle are art treasures in the form of Pendants and Jewels. There are many chests which may also contain jewels, to open these chests simply shoot at them, the screen prompt at the bottom of the screen will tell you it's contents when opened. A lift pass can be found in each level in order to use the lift to get to the other floors. You'll soon notice boxes of explosives laying around avoid shooting at these at all costs, otherwise your game will end. Along with the explosives is a detonator which must be set off on each floor in order to complete the game.

EAGLES NEST continued

Into The Eagles Nest is a very good arcade adventure, with plenty of playing depth. Rooms can be entered in any order so strategy plays an important part. The graphics are sharp and colourful, however, lacking in good sound effects. Effort has been made to include a high score table where entering your name is done via the joystick. Overall I was really impressed with the game though for the price of £17.50 it may be advisable to try before you buy. Into The Eagles Nest is currently available from Micro Discount at £17.50. Unless you know a cheaper source ?

FRED.

REVIEWED by Mark Fenwick.

Fred from Avalon Poland, distributed under licence by Zeppelin games.

Fred is a game which revolves around Fred, a caveman who wishes to find the meaning of life. Well, the life of a caveman anyway.

On booting Fred you find yourself looking at a well put together title screen. Here we see our caveman friend armed with an aerosol and surrounded by nasties, crabs, frogs and birds to be more precise! Towards the bottom of the screen is Zeppelins scroller converted nicely from Polish so it's easily read. A pleasant sound track accompanies this which continues throughout the game. From here, pressing fire takes us straight to the action.

The playing area takes up the top two thirds of the screen while the lower part shows score, hi-score, lives left, distance accomplished and objects picked up so far. Our caveman has come equiped with weapons in the shape of rocks (now, now, enough of that...) which soon dwindle so other weapons can be picked up as well as more rocks. Fred can be seen wearing a typical dress of a bearskin as well as other detailed features. The play area shows levels of rock and cacti and is colourful and well detailed. Movement is achieved via joystick, to jump on to another level simply move the joystick at a diagonal. To throw rocks press the button, which will also activate the aerosol nasty killer later on.

The game scrolls smoothly from right to left at your own pace as you reach the end of the present screen it will automatically scroll. You'll soon see the nasties which can easily be killed by a carefully aimed rock thrown by Fred. If there's a nasty on a level above then you can throw a rock while jumping so as your path is cleared. Should you touch a nasty then you lose a life and return to the beginning of the particular level you are on. As you do move through the level you'll notice a flashing dot on the distance bar at the lower of the screen. You'll soon find that there are plenty of levels to go at. To pick up more rocks simply walk over them. Once you have ten a symbol of three rocks will appear at the bottom of the screen. You'll also come across what looks like an early Roman jug. This hides either special functions and weapons or just another cacti.

FRED continued

To see the hidden item simply throw a rock at the jug to reveal its contents. To pick up the item simply walk over it, it will then be dispayed at the bottom of the screen. Up to eight items can be carried at any one time, any after eight will be lost. There are various things to be picked up, there's a Hat, Shield, Aerosol, Welly, Rocks, Extra Lives and then there's the cacti Which should not be touched. Make sure you pick up the item fairly quick otherwise it will disappear. The Hat will give a limited time of cloaking so if you touch a nasty or a cacti you won't lose a life, it will however depreciate each time you touch something. The Shield will give the same protection but only lasts a short time no matter what you touch. The Aerosol will fill the bottom left hand corner aerosol with nasty killer, the aerosol level will depreciate with each spray. Each time you now press the button Fred will spray his enemies, any more than one can carried will be displayed at the bottom of the screen. Rocks will be shown as only three but is infact ten. On the later levels where sulphur pools appear, the Wellies will come in handy and will protect Fred from death. It's worth remembering that although you may have a full collection of goodies, you should tread carefully as death will result in losing them all.

The levels progress from very easy to hard as at later levels there's a lot more nasties to avoid as well as traps with no way out. At the end of each level a twenty second bonus is given in which time you should try and grab as many goodies as possible before entering the next level.

Fred is really enjoyable to play, very leisurely with no silly time limits, so you can just stroll and time your jumps under no pressure. What with the concept, gameplay, superb graphics and music this is truly a worthwhile game to play. Fred is available from Micro Discount for £4.95 and well worth it too!

SOLUTION to VOODOO CASTLE

Voodoo Castle is an excellent adventure by Scott Adams' wife under the Adventure International flag.

In order to solve the adventure you must free the Count of Monte Cristo from a curse by using all your cunning and puzzle solving talents. In the Page 6 golden oldies column the adventure received a two star rating and was noted for being of limited vocabulary and having illogical puzzles. Obviously the reviewer was not a true adventurer otherwise he would have known that we lamp-holding, map-reading die-hards have twisted minds and love nothing more than an illogical brain teaser! Don't we?

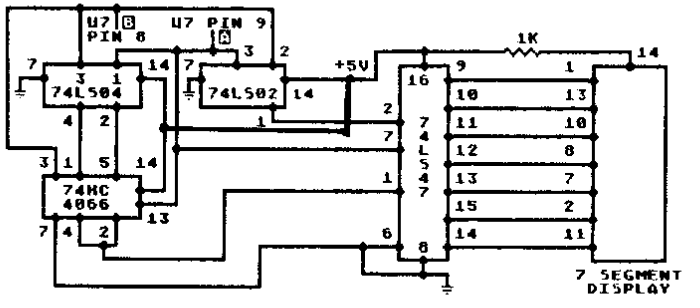
Open coffin - get ring - E - get knife - wave ring - drop ring - go chute - get plaque - go hole - summon medium - look ball - W - W - go fireplace - get idol - S - E - S - W - S - E - get sword - get shield - W - N - E - get glass - E - move kettle - go hole - read plaque - drop plaque - clean idol - get foot - U - N - pull head - drop sword - turn 38 - turn 33 - get hammer - drop glass - drop head - E - E - mix chemicals - get chemicals - eat chemicals - W - W - S - W - W - S - go door - look grave - get clover - get saw - E - N - E - N - N - go window - get doll - S - S - put foot - on man - W - go fireplace - open flue - go flue - pull nails - drop nails - drop hammer - pull boards - saw grate - drop boards - press button - push sweep - read paper - drop paper - D - D - S - E - S - go stairs - say Zap - listen - D - E - N - E - E - get bag - W - W - S - W - N - drop shield - E - go chute - wave bag - go crack - get page - S - open bag - get stick - get book - read book - read page - drop book - drop page - drop bag - go hole - look ball - W - circle coffin - wave stick - yell chant

Provided for TWAUG by M. Stinson.

CORRECTION TO SEVEN SEGMENT DRIVE # INDICATOR DIAGRAM.

Pins 14 on Chip numbers 74HC4066 and 74LS04 are connected to +5V as drawing below.

DISCLAIMER: We would like to make it clear that we, T.W.A.U.G., cannot be held responsible for any damage caused to your hardware through using any material published in our newsletter.



WHY AM I DOWN ON IBMs?

by John Kasupski, WNYAUG
Reprinted from WNYAUG POKEY NL 11/91
by THE OL' HACKERS with THANKS!

This article is re-printed with permission from "THE OL'HACKERS" disk newsletter issue #12. We thought it would be a shame for our newsletter subscribers, who do not read the OHAUG newsletter, to miss this amusing article.

A few weeks ago, while reading through the message bases on 'The Wizards Attic', I saw a message posted by an IBM user who was having trouble with a file compression utility. A few Atari ES users (myself included) left replies such as "That's what you get with an IBM" and so forth. The IBM user replied in kind. After pointing out the difference in clock speeds between the Atari and IBM CPUs, he declared that using an Atari was "remaining in the dark ages of computing." He wanted to know why we were "so down on IBMs". I feel his question was valid.

I AM down on IBM products. I have no desire to own an IBM or compatible. I make no bones about it. But why, indeed? To begin with, my Atari system sits on a modest desk and hutch in a small room off the kitchen. It's a HOME personal computer, exactly what the Atari was intended to be from the moment it was created: "The Atari 400 Personal Home Computer". The rest of the Atari ES computers followed in the footsteps of the 400 machines. But, guess what the initials IBM stand for? International Business Machine. Not a home computer, not a personal computer, a BUSINESS MACHINE

You know that Every single solitary IBM computer ever sold was designed for the business market, except the PCJr., to which the buying public quite properly extended about as much welcome as a fart in church. Yes, an Intel 80386 is faster than my Atari's 6502, but if speed were the only consideration I'd go out and buy a Cray mainframe and set it up in the basement, put a terminal in every room in the house, do all programming in ADA, and spend the rest of my life making payments on the darned thing. The IBM as a home computer? Suppose you do purchase, say, an 80386 machine with a clock speed approaching Mach 1 and a price tag to match. It'll cost you around $2000, with monochrome monitor and NO graphics unless you shell out an extra $500 for another card to plug in to the motherboard. This is only the beginning. Every time you want to connect something to your computer you'll have to buy ANOTHER expensive add-on card. OH, you just want to get the basic system running? Not yet: The whole pile of hardware you bought is WORTHLESS without a DOS! That'll cost you another $100 or so. Never mind that MS-DOS is the most user-hostile DOS ever invented, because soon you'll discover that your IBM MS DOS BASIC does no syntax checking while you're entering a program (you don't find out about syntax errors until you try to RUN your program). And although you CAN set up strings of ANY size, string FUNCTIONS won't work on strings that are longer than 255 characters.

You don't want to program, you just want to do word processing? Better get some more memory first!

ATARI versus IBM continued

On my Atari, a GREAT word processor, TextPro 4.56xe, takes about 18K of RAM. (It's ShareWare, the requested donation is about $15.) On your new IBM, even a mediocre WP will eat up six times that amount of RAM, while a good one like Word Perfect (about $300) is so large it REQUIRES a hard drive. Incidentally, IBM dealers will SWEAR you CAN'T connect a hard drive to an Atari ES. The 40Mb hard drive now connected to my 800XL didn't believe them; neither should you.
(EDITORS NOTE: He is right I too have a 10 meg Hard Drive, as do most BBSs)

Another reason why an IBM does not strike me as a practical choice for a home computer: No matter how fast my CPU is, the signals are still going to come through my modem at 1200 bps, and I am still going to type at the same 20-30 WPM that I always type at. Why pay through the nose for a CPU that is going to spend half of its cycles sitting idle, waiting for me to do something?

But that's only the hardware half of the story. What about software? Why is the manual for MS-DOS as thick and heavy as a telephone pole-and about as easy to understand? The biggest manual I've seen for an Atari software product is for Kyan PASCAL (a complete programming environment).

It explains how to write and compile PASCAL source, how to use the KIX Environment, the whole ball of wax. On an IBM you need a manual twice that size just for the DOS alone (you can forget about using your IBM until you've absorbed the manual). On an Atari, you can read through a few pages of documentation and start being productive. Why? Because the computer was intended for home personal use, so the software packages are geared to the home personal computer user. You can teach yourself to apply Atari software. Business schools charge money to teach you how to use the IBM. Come to think of it, all this got started with an IBM user having a problem using a file compression utility. I've never had any problem with the Atari, in fact I have as much fun running the UNARC utility as I do using the files I UNARC. The very first time I called a BBS, I needed to upload to improve my status. I logged off the board and proceeded to ARC some files, successfully, on the first try. I uploaded the next day. A few days later I called again and downloaded an ARC file. I successfully UNARCed the file on the first try. It was an Atari board running BBS Express Professional. Some time later I called an IBM board that was running WWIV. I couldn't even FIND the files databases, let alone attempt to upload or download anything. I couldn't even get the user help files to capture those! I decided that if this was a sample of what IBM software was like, I wanted no part of it. Nothing in my experience since, including actually using an IBM compatible, has done anything to change my mind.

That's why I'm down on IBMs. I won't go so far as to say that I wouldn't take an IBM machine if someone offered it to me for free, because I would.

ATARI versus IBM continued

But, I'd immediately sell it so I could buy more Atari stuff - like a bigger memory upgrade, or a 2400 bps modem, or a second phone line so I could put up a BBS, or...well, the wish list goes on and on, but it will never include an IBM. The REAL "Dark Ages Of Computing" began when IBM managed to permeate the business market with its products, which allowed IBM to convince a large segment the public that IBMs were the ONLY computers worth buying. Never mind that these machines don't belong in the home environment to begin with: The corporate culture of IBM execs, replete with two-hour long, three-martini lunches, has a price. Somebody has to pay for it, and it may as well be the buying public! I, for one, am not fooled. I'll buy my own lunch, and my own home computer - which, until they start moving animals in pairs to Cape Canaveral, will be an Atari ES!(8 BIT MICRO)

OL' HACKERS EDITORS NOTE:
No way could I have said it better***
NOW BE VERY HAPPY THAT YOU WERE SMART ENOUGH TO BUY AND STICK WITH THE 8-BIT

GUARDIAN of the SWORD

This game play part is missing on the issue #2 disk for the adventure game, and if you still cannot solve the puzzle use the solution at the end.

As a not-yet-famous archeologist, you have stumbled on some ancient maps, long buried in a museum archive. The maps show an old church which did not not know existed, and they show reference to great burial in 539 AD.

After finding the church, you are convinced it has something to do with the legend, and you are determined to find out.

The game is played with a verb-noun input, and has a reasonable list of verbs and nouns. All the items found in the adventure are required to solve the puzzles in order to complete it.

Games may be saved to cassette or disk, with the save command. Movement is achieved by using N,S,E,W,U,D.

My only hint is that a few items are worth examining or reading.

SOLUTION

Read text from bottom right to top left - go north = htron og

ECILAHC EVIG-RITNALP LLUP-W-S-S-LLAW KCIK-W-W-W-S-S-KNARC NRUT-KNARC LIO-D-D-E-N-YEK PORD-ELTTOB PORD-ECILAHC LLIF-ELTTOB EKAHS-NAC EKAT-ECILAHC EKAT-TSEHC NEPO-SEHCTAM PORD-ARBALEDNAC THGIL-SEHCTAM EKAT-ARBALEDNAC EKAT-MLASP YALP-REVEL LLUP-N-ENIW EKAT-SDRAOB KAERB-N-N-E-E-S-EDAPS PORD-SSORC EKAM-S-YEK EKAT-GID-N-E-EDAPS EKAT-W-S-S-DOOW EKAT-HCNEB KAERB-S-E-GNIRTS EKAT-ETAG ENIMAXE

PRINT SHOP utilities INFO

We had a number of enquiries about Print Shop utility programmes in our library, what they are used for. From now on we will give a short description of a few files in each issue. The TW000 and the three numbers after it is as they appear in the PD Library catalogue.

TW026: This utility is an Atari Basic program which converts Print Shop icons for use with the 'Newsroom' program. The converted icons are written as "Newsroom Photo" (not as Clipart).
The program is menu driven, it offers the following options:
D - Toggles the destination drive between 1 and 2.
V - Toggles the write verify function ON/OFF.
S - Togles the scaling factor between single and double for icon conversion.
C - Determines how many photos can be written to a given disk.
CR - Carriage return to start copying.
X - Exit the program (to the Basic editor).
Documentation is on the disk as "PS2NEWS.DOC". You can use the 'C' option from DOS to print this file, it is set in 40 columns.

TW027: This is a fine label printer program, Print Shop icons can be printed on the label and a large number of fonts are also on the disk to use for printing. This is one of my favourite labeller.

TW028: This disk contains five different programs to use with Print Shop icons.
You've got the option to greate your own graphics and fonts. You can convert PS pictures to visualizer format or Micro Painter pictures.
There is an option to let you view nine PS pictures and their names at once. You can also save them as Micro Painter picture and dump this to a Seikosha GP-100AT or on Epson printer. You can greate a slide show and show any Micro Painter picture. It also converts Screen Magic files to DOS 2.0 compatible files. (Micro Painter format). A help file is also on the disk.

TW029: This is a handy PS program, it is easy to use and is menu driven. With this program you can format disks to PS format or Atari DOS format. You can transfer Print Shop icons to Atari DOS format using one or two drives. Make a hard copy of your PS icons, read and print out the directory, rename or delete any icons on the disk.

TW030: This is another PS utility which is menu driven with 8 options. Lets you read the directory to screen or printer. Prints Graphic icons to printer, format data disk, configure utility disk and rename data files.
When printing to printer make sure line spacing on your printer is set to off, this utility has its own setting.

TW031: Load this disk with Option pressed, you will see the Bellcom menu on screen when loaded. Use arrow keys (no CTRL) and highlight DUP.SYS press return, call directory and use 'L' option to load OBJ files. To unpack PACK files use GAGPACK.OBJ file, to view choose SHOPTOOL.OBJ file. Use seperate disk for unpacking files.

DISPLAY LIST INTERRUPTS

by Nir Darey

Before you go on reading about DLI's, you must know all about the DISPLAY LIST of the ATARI 8-bit computer. There is a good article about this subject in the "NEW ATARI USER" mag. ISSUE #51 from Ian Finlayson.

Now let's start explaining what DLI is. The television draws the screen image in 20 miliseconds, at this speed the computer has plenty of time to change parameters while the display is being drawn,

The computer responds to the request interrupt that you put in the Display List, changes parameters and returns to it's normal business. Of cause you shouldn't do to many things because it affects the time the screen is drawn.

That's enough theory let's talk business, The registers for the Display List Interrupt are in 512,513 you should insert in them the address of your routine. Always remember to enable the DLI, by setting the NMEIN register (54286) to 192. otherwise your routine will not be executed!. Now tell the ANTIC chip at what line it should execute your DLI routine by setting bit 7 at the desired line on the DISPLAY LIST.

There a are few things you must always remember when writing a DLI routine:

1) Make sure you restore the Accumulator and the X and Y registers in the stack if you use them, and at the end of the routine restore them back.

2) End your routine with RTI (Return From Interrupt) instruction.

3) When using graphics registers (colors,scroll,character sets etc...) also store the values in the WSYNC (Wait For Horizontal Sync) register, so the next command won't be executed until ANTIC finishes drawing the current scan line. By doing this you prevent any flickering on the screen.

4) Make sure you change the hardware registers, not the shadow registers.

Now let's look at a small DLI routine, the following source code was written in MAC65.

```
10      *=1536
20      PLA
30      LDA 560  ;those 4 lines
40      STA 203  ;finds the address
50      LDA 561  ;of the
60      STA 204  ;DISPLAY LIST
70      LDA #<DLI
80      STA 512
90      LDA #>DLI
100     STA 513
110     LDA #192
120     STA 54286      ;enable DLI
130     LDY #15        ;lines 130-170
140     CLC            ;sets bit-7
150     LDA (203),Y    ;at line 15 of
160     ADC #128       ;the
```

DLI continued

```
170     STA (203),Y    ;DISPLAY LIST.
180     RTS
190 DLI PHA
200     LDA #40
210     STA 53272   ;COLPF1
220     STA 54282   ;WSYNC
230     PLA
240     RTI
```

The next program is for the basic user.

```
10 ADDR=1536
20 READ DAT:IF DAT=-1 THEN 50
30 POKE ADDR,DAT:ADDR=ADDR+1
40 GOTO 20
50 X=USR(1536):END
70 DATA 104,173,48,2,133,203,173,49,2
80 DATA 133,204,169,36,141,0,2,169,6
90 DATA 141,1,2,169,192,141,14,212,160
100 DATA 15,24,177,203,205,128,145,203
110 DATA 105,128,145,203,96,72,169,40
120 DATA 141,24,208,141,10,212,104,64
```

If you want to use more than one DLI in the same program, (let's say 2 DLI'S) you must set in the first DLI, the second DLI address in the DLI registers (512,513) just before you restore the processor registers. and at the second DLI routine you must point back to the first. Make sure when using more than one DLI, you enable them (remember storing 192 at 54286) during VBI, otherwise they wont executed in the right order. Don't forget to set more than one DLI request at the DISPLAY LIST.

For more information about multiple DLI's read Eisbaer Corp. article at "MEGA MAGAZINE 4", and more technical information you can find in "DE-RE ATARI" book.

Next time I will write about the SOFTWARE TIMERS and how to use them.

DISK CONTENT PAGE MISSING.

Due to lack of space the Diskcontent page is missing from the newsletter, but here is a short list of the programmes you can find on the disk.

On side A you have: Topshelf the instructions are on page 19. Wordweave the doc is on page 20, this is a handy program to write adventures. Diskjacket maker, this program lets you print the outline of a diskjacket onto A4 paper. Three games are also included: Mazewar, Firebug and Race in Space and three little demos. Use the Unarc file on side A to unarc the arced file on side B. The arced file is a 24 pin printer driver file, you will also find a 24 pin colour printer driver for the Citizen Swift on side B.