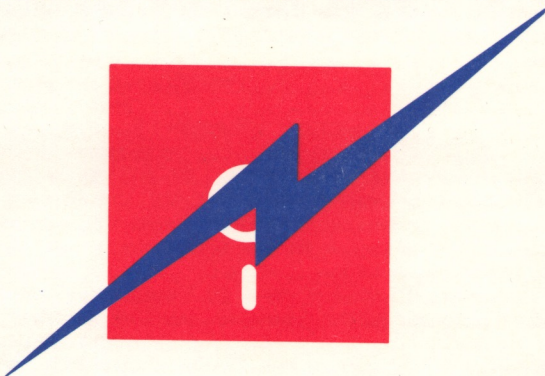# FOR ATARI*
400/800/1200

diskwiz-II

**COMPLETE & AFFORDABLE
DISK EDITING REPAIR & DUPLICATION
SYSTEM FOR ATARI COMPATABLE DRIVES**

• single load • fast mach. lang. • repair, elplore, dup dos/non-dos sectors •
simultaneous hex/ascii display and editing • print out all modes to any
printer • dumps special & inverse chars to EPSON*, NEC 8023*, PRO-
WRITER*, GEMINI*, M-T'SPIRIT* • fast mapping and byte searches • file
link trace • speedcheck and adjust • block move • auto link pointer, file
code change • vtoc bit map changes or check • cross sector disassembler
• fast/slow copy • 1 or 2 drives • hex-dec-asc conv. • complete manual •
create ''bad'' sectors • fix deleted or open files • fix dup filenames • safely
use non-formattable disks • single and double density.

ALLen
Macroware

*Registered Trademark

diskwiz-II

by Jerry Allen
copyright (c)1982,1983
Allen Macroware
POB 2205
REDONDO BEACH, CA 90278

ATARI is a trademark of ATARI, INC. and all references to ATARI or their products should be so noted.

Backups are available for $10 ppd. when returning enclosed warranty form.

LIMITED WARRANTY

This program is intended only for legal repairs, alteration, and duplication (for archival purposes) of program materials stored on disk.

Please direct any comments, suggestions or questions along with a stamped self-addressed envelope to:
Jerry Allen
pob 2205
REDONDO BCH, CA 90278

THANK-YOU FOR BUYING THIS PRODUCT.

diskwiz-II

TABLE OF CONTENTS

CAUTION!

This program allows you many freedoms not allowed by DOS. This freedom can be very powerful, and, it can be equally as disasterous. The distributor , author, and all others involved with the manufacture and sale of this program can be in no way responsible for problems resulting from the use of DISKWIZ. DISKWIZ is merely a tool similar to a wrench, hammer or saw which can also be of great harm if used unwisely.

Please keep the following in mind when using this program:

1. Always duplicate a prized or valuable disk prior to attempting changes or repairs--the dup function on DISKWIZ will dup any good data on any disk

2. Always keep the write protect tab on originals--remove only as a final step and replace immediately

3. Plan moves and data changes thoroughly prior to beginning the actual work

4. Make your changes first to the duplicate

5. Use the map, trace, and VTOC functions frequently to determine potential problems

6. Keep blank formatted disks at hand to save new work

7. Only make changes to originals after working out changes on another disk and satisfying yourself that your new code or data is correct

LOADING DISKWIZ

Boot up with your BASIC cartridge installed. Although DISKWIZ uses machine language routines for all space and time critical functions, it still accesses these functions through use of the BASIC cartridge, for security and other reasons.

If the READY prompt appears after the disk has seemed to finish loading, then the program has not properly loaded.

The buffer (memory storage area) size for sector transfer is based on your computer's memory size. The buffer for a 48k machine will be approximately 196 sectors per pass. In a 32k machine about 76 sectors. In 16k machine about 4 sectors. So, if you are using a 16k machine, which means you don't have a second disk drive, be prepared for a lot of disk swapping during duplication of disks and other functions which allow you to exceed the memory buffer size, and require a write to another disk. Some functions, such as R and E, if rewriting to the same disk will not cause this inconvenience when cylcling through the buffer. And other functions are not buffer dependent at all.

SRESET AND RETURN

SRESET can be used to abort your mission after you have determined that you have made an error and have no other alternatives to back out. DISKWIZ will restart automatically.

RETURN is used to leave many functions like, F>FORMAT or the H>HEX-DEC-ASC CONV. It can also be used to answer "no" to commands which prompt for a "y" or yes. A RETURN in answer to a prompt requiring a numeric or string entry causes an error and the program returns to its main prompt, "ENTER SECTOR# OR COMMAND". This may be more desirable in many cases then a SRESET abort.

THE MAIN DISPLAY

This display shows all 128 (or half of a double density sector) bytes of a sector along with other information. In double density, pressing RETURN will toggle the display between both halves of the sector being displayed. On the far left is a column of hex numbers which are the first numbers of that row of bytes in the sector. (This is a typical hex dump). The row of numbers across the top of the hex field shown alternately non-inverted and inverted, are to find the last hex digit of the column you are looking at. The inverted characters are for all rows starting with a last digit of 8; the non-inverted are for rows starting with a last digit of 0. The hex field columns in the middle correspond directly to the eight columns of ATASCII characters which are shown on the right. The information at the bottom generally applies to DOS2/OSA+2 format sectors and all except sector numbers and status can be ignored for non-DOS/OSA+2 type sectors. The status codes are the same as the inside cover of your manuals. Status 1 means 'ok', 144 'bad sector' or write protected disk. In DOS type sectors, the "next bytes" tell DOS which is the next sector of the file. The "file code" byte is used by DOS to make sure that all the sectors it is loading are from the same file (error 164 if not). These codes relate to entries into the directory which is, except in rare cases, at sectors 361-368. "Used bytes" tell DOS how many bytes in the sector to actually use. See APPENDIX B for pictorial aid.

DISKWIZ can be used to alter any format. DOS1, FORTH, etc. See your specific users manual for format details, and/or study disks to determine format requirements.

A>ALTER DATA

This command is used to randomly change any byte or group of bytes within a sector.

First, you must call up or have the sector you wish to

change on the screen. Remember to RETURN if you are in
the directory format. Type "A", RETURN. The first prompt
asks which field, hex or char, you wish to edit. You may
also choose to exit at this time by hitting RETURN.

Once you have chosen which side of the screen to work
with, you will be limited to that side until you have
edited a line and the initial prompt appears again. You
may only edit one line at a time. After you respond to
the question of which field to edit, the '?' and cursor
will appear at the upper left corner of the field you have
chosen. Use the CONTROL and ARROW keys to move the cursor
to the location you wish to edit (all editing keys, and
wraparound still work, so you can quickly jump from the
top to bottom of the screen). Edit one or a whole line of
bytes. You must press RETURN while on the line you intend
to edit, or the change will be ignored. The program will
recycle and you can change fields and continue or exit to
the W>WRITE routine. No change to the sector on disk
will actually occur until you deliberately write the
screen changes to the disk. If things are not right or
you are unsure of the correctness, exit, answer negatively
to the write prompt, call the sector back up, and start
over.

Exits may be accomplished in a number of ways: At the
first prompt , hit RETURN; when the cursor is at the top
line next to the ?, hit RETURN; a RETURN pressed on any
blank horizontal row; or, at last resort, hit SRESET.

IMPORTANT!--Only one thing must be kept in mind
when editing the CHARACTER field: An inverted escape
character ($9B,155), or line return character, anywhere on
the line you are editing will cause itself and all
following characters to become spaces ($20,32). The
cursor will go directly to the hex field edit mode in this
situation. The first space on the line just edited is
usually where the the return character was, and you can
replace it at this time by putting "9B" in the hex field
at the corresponding character location. If the characters
following the 9B were not intended to be spaces ($20,32),
there are two ways to avoid this problem:

The first, and most difficult, is to only edit lines with
a return character ($9B) that needs to be retained using
the hex field mode only. The second method is to use the
character field mode and change the inverted escape return
character to an asterisk or some other dummy byte you wish
to use. Edit the rest of the line, and then return to the
hex field and change the dummy byte back to "9B".

See the H>HEX-DEC-ASC CONV for aid in using the A
function.

See the R>RENUMBER section, and RENUMBER first if you
intend to move your altered sector and still retain the
old one.

D>DUP SECTORS

This command is used to transfer sectors from one disk to
another.

At the prompt, enter the consecutive first and last sectors you wish to dup. This may be of the range 1,1 to 720,720 or 1,720. Press RETURN.

The first prompt asks if you intend to write from drive#1 to drive#2. If so, both drives must be of the same density. If you have or intend to use one drive only, press RETURN. At the prompt for the destination disk, press RETURN to use drive#1, or enter "Y" to write to drive#2. If drive#2 is selected, the program will cycle to completion of the request. If drive#1 is used and your buffer size is exceeded, the prompts to insert source and destination disks will continue until the range of sectors you specified has been transferred.

"Bad" or scrambled sectors will be dupped as all zeros usually. Sometimes they will be readable, but your new sector will not generate the necessary error code required by the program before it will run. Use the M>MAP function or specified sector entry command to locate "bad sectors".

E>ENTRY:DIR. FILE

Use this command to alter the directory file entry code byte, $7D(125).

There are 64 possible codes used in this program. These correspond directly to the 64 possible directory entries. An "A" code is entry #0 or the first entry in the directory. A "B" would be the second or entry #1, and so forth. All DOS created files use this code to check the integrity of its files. If DOS encounters a sector with a different code or the filename you specified for a load does not match with the directory entry, an error #164 will be generated. This applies to all DOS files: BASIC, AUTORUN.SYS, and other object code files. Please note that the code used in DISKWIZ is not the same as DOS and is used for space requirements in mapping and entry ease. Also note that non-DOS files generally do not use link pointers and file name codes.

The E command will change all codes in the range you specify between one and 720. Check your directory before using this function. If you are entering a new name in the directory be sure to use a deleted or the first all zero entry space. If your new entry skips an all zero entry space, the file will be ignored by DOS. Then use the file code from the entry line you just used in the directory to enter at the prompt when the E command asks for your new code.

The formula to manually enter byte $7D(125) is the entry # in the directory (starting at zero) times four plus the high bits of the link pointer (0,1,2). This command as you can see greatly simplifies this procedure. It will cycle to completion for one or any number of consecutive sectors that you specify.

See the F and V commands and APPENDIX B,C,D.

F>FORMAT:DIRECTORY

The F command changes the normal sector display to a special format for reading your directory.
When a normal directory number is entered (361,368), enter F, RETURN to change the screen format. Hit RETURN to toggle back to the normal display mode. You may toggle back and forth between the two displays by alternately entering F, RETURN, and RETURN only. The F command to a non-directory sector will give unpredictable results. The directory location may be changed in DOS to hide it. If you find this to be true on a certain disk, the F command will again be useful.
When changing your directory entries, toggle back and forth as described above to read and change the sector.

G>GRAFIX OPT

This function allows you to copy all ATASCII as it appears on the screen, allowing you to leave your terminal with a written record. You must have an EPSON MX with GRAFTRAX (or EPSON compatable), or a NEC/PROWRITER printer for this function to work. If you don't meet this requirement for the printer, use the P>PRINTOUT command.
With the normal sector display, enter G, RETURN. In the map or trace modes, you may also copy the screen by pressing the G (or P) key while the display is still up. You do not need to hit RETURN also in this case; and the normal sector display mode will reappear after the printer finishes.
See U>UTILITY MENU to configure the program for EPSON/GEMINI type printers to NEC/PROWRITER type printers.

H>HEX-DEC-ASC CONV

This command gives you access to an on board conversion calculator.
The first time that you enter the H routine, the cursor will just sit and wait for your first input. Enter a range of numbers from $00-$FFFF or decimal 0-65535. Also, entering only a letter or other character will return the hex number for that ASCII. Entry input errors will return the ASCII in hex for the first character in the string entered. For hex values, you must preceed the entry with the traditional "$". Two ASCII characters corresponding to the numerical answer will also be displayed.
Entering RETURN only, exits this mode. Use normal editing keys to change or clear your input line.

L>LIST COMMANDS

This function re-displays the command menu and allows return to the previously viewed sector.

M>MAP DISK

This function reads and displays a code for all sectors, 1-720, simultaneously on the screen.
When the M, RETURN, command is given, a prompt will appear asking if you want to map just a part of the disk. RETURN only will default to sectors 1,720. The screen will start filling with sector codes which were defined at the last prompt.
Error and zero codes apply to all disk format types. File name codes and inverted character (in use) codes only apply to DOS files. Due to screen size limitations, you will only see sector numbers at the beginning and end of lines 30 increments appart.
Use the T>TRACE command to trace a single file by link pointer from beginning to end. This will apply to both deleted (non-inverted) and "in-use" files whether or not they can be found in the directory. Using the trace function will also help to know which sectors to ignore since they are not part of the DOS file, but still generate the same code in their last three bytes. Due to the fact that DOS reuses a file entry space of a previously deleted file there may be many deleted files with the same file code, but they are still not related. These files will all be non-inverted on the map. See the section on recovering a deleted file.

P>PRINTOUT

This command is the same as the G command except that all non-printable characters on the printer will be changed to periods or spaces. All inverted letters from the screen will be printed as non-inverted. The hex codes will tell you whether or not the character is really inverted or not and what the period really stands for when viewing the normal sector display.
This function is the same as The G command in respect to the map and trace functions' screens.
See the G command and APPENDIX: F

R>RENUMBER

This command is used to change the location of a sector or group of sectors on a disk.
The first prompt asks for a range of sector numbers. This may be one sector (i.e.,"134,134") or a group of sectors (i.e.,"220,550"). The second prompt requires only the first sector number of your destination. The program will

determine if your last destination number is out of range.
You must determine if you will be overwriting necessary
sectors or are out of range before you begin.
If you are going to overwrite something you'll later need,
use another disk for your destination disk when prompted
to do so.
A special message will appear telling you to definitely
use a second disk if the first sector of the group of
destination sectors is between the first and last sectors
of the sectors you are coming from, AND, the amount of
sectors you are moving exceeds your buffer size for one
pass. If you don't comply, there will be an overlap
ruining the beginning of the next pass. A second disk is
not mandatory if this condition does not exist.
You may use drive#2 for writing to the destination disk
when prompted.
After moving DOS created sectors remember to use the
S>SECTOR LINK PTR and E>ENTRY:DIR. FILE functions to
revise the link pointer and file bytes as necessary.
This command as most others, except the T>TRACE command,
is consecutive in nature. Therefore, to change a file
that is not consecutive on disk into a consecutively
stored file, you must use the T>TRACE function to locate
all the parts of the file, and move them as groups to one
area. An example follows:
Move a file from 50-60,100-120,402 to sectors 250-282.
First renumber 50-60 to 250. The last sector used at this
time would be 260, which would now be on the screen. This
sector plus one (261) is the next destination sector for
100-120. After this move the screen will be at 281 which
is now the new last destination sector. So, you add one
to that (282) for the next destination sector, and move
the last group which is only a single sector in this case,
to this destination. Now, the S command and E command,
if necessary, will painlessly revise your link bytes.
Also remember to change the two start bytes in the
directory to point to 250 instead of 50. They are the two
bytes immediately before the filename and are of the
typical hex low byte first configuration. Toggle the F
command and use the H command for help. Use the A
command to do the job.

S>SECTOR LINK PTR

Use this command to refresh or revise a consecutive group
of link pointers especially after a move (R>RENUMBER).
This command works in a sequential manner only.
Assuming the file to be changed is sequential (located in
one group), merely enter the beginning and ending sector
numbers at the prompt. DISKWIZ will automatically pull
all the sectors you specify, change the pointer bytes as
necessary, and rewrite them to disk. The last pointer
must be specified in response to the second prompt. If the
last sector is the last sector of the file, the last
pointer will be zero.

In the case of jumps, use the trace command if you have
not already determined the breaking points. Use the S
command on each separate group. At the last sector of
each group of the file the pointer must be to the first
sector number of the next group or single sector which is
next. The last sector of the last group should be zero.
Adding sectors at the beginning or end of the file is
handled similarly.
[The math of it all: The link pointer bytes are at bytes
$7D(125) and $7E(126). Byte $7D(125) is a composite of
the file entry code and the high bits of the link pointer.
Since there are only 720 possible sectors, the high bits
can only be 0, 1, or 2 (*256), and are stored in the two
lower bits of byte $7D. To come up with the byte, divide
the sector number to go to by 256. The integer portion
(0,1,2) is inserted into byte $7D. (The upper five bits
are the file entry code times four.) The sector number to
go to, minus the number you just calculated times 256,
equals the low byte of the pointer and is stored in byte
$7E(126). Needless to say, this command saves you a lot
of time and agony.]
Use the T command to find breaks in bad files and use
this command to repair them. Sometimes a file can only
be partially recovered, but that in itself may be of help.
There is no need to consider the file entry code bytes.
They will be handled automatically.
See the R and E commands and the APPENDIX.
Many non-DOS created files are linked sequentially or by a
table in the program and do not have link pointers, file
entry code, and bytes used bytes. These include many
commercial programs and all boot sectors. Also, the
directory and VTOC do not have these bytes. Do not
attempt to use this command on sectors of this type.

T>TRACE

This function will trace through a DOS linked file from
the sector you specify to the end. The end will be a
sector with a link pointer of zero or a link pointer of
greater than 720.
This function is not necessarily sequential in nature as
are many of the other commands; and, is almost mandatory
to use prior to or in conjunction with some of the other
functions. Although the display looks similar to the
M>MAP DISK function, the VTOC in use designation is not
implemented here. (See the V command for VTOC spot
checking). The hearts mean nothing but are left to count
with.
The last sector found will be shown when the normal screen
returns upon pressing any key after the trace is through.
A printout of the trace can be made if a printer and
interface are on line and the P key (or G if
appropriate) is pressed. The print will take place, and
the normal display of the last sector will then appear.

diskwiz-II

U>UTILITY MENU

This command displays a submenu allowing you to change
density, initialize a disk, change the grafix printer
type, or turn write verify on or off when writing to a
disk.
Enter the desired number of the function you wish and
RETURN. The initialization command only writes track and
sector information and erases a disk (you cannot write to
a fresh un-initialized disk). It does not write sectors
one thru three and 360-368 as DOS does. If you need those
sectors, copy them later from a DOS formatted disk. The
change from single to double density only reconfigures the
program. You must manually reset the drive to double
density by setting appropriate switches and/or turning the
drive of then on as specified in the drive manual. RANA
drives: refer to manual; PERCOM: set switch #4 on rear of
drive to on, then turn drive off then on again. NCT TURBO
810: turn drive off then on again. Refer to drive manuals
for assistance. In most cases the operation may be
reversed to return to single density.
The write verify choice allows slower but slightly more
reliable writes.

V>VTOC

This command allows the user to check, lock, or unlock the
bits of the bit map sector, or Volume Table of Contents.
DOS uses this map to determine which sectors it can or
cannot use to write to.
The VTOC is located at DOS/OSA+ formatting time at sector
360, just before the directory sectors. There 90 bytes in
this sector which DOS uses to keep track of the disk.
This works out to one bit per sector, or eight times
90=720. The 90 bytes start at byte $0A(10) of the VTOC.
The highest bit of byte $0A, bit 7 ($80,128), represents
sector zero; the next bit, bit 6 ($40,64), represents
sector one; and so on down to sector 719 in byte $63(99),
bit 0 ($01,1). (Sector 720 was somehow ignored when DOS
was written, while sector zero was acknowledged even
though the disk drive controller cannot write to sector
zero, but can to sector 720.) If a bit is on, or one, the
sector is available for use; off, or zero, tells DOS that
the sector is in use.
If you have added or deleted any sectors with this program
to a DOS formatted disk that you intend to use again with
DOS, you must use this command to alter the VTOC or DOS
will overwrite your new sectors and/or will not be able to
use sectors which you deleted. (DISKWIZ does not use DOS
as you may already know).
The VTOC command takes care of all the tedious
calculations and corrections automatically. The range
specified at the prompt will be sequential. The check

function only works for the sector you are presently
viewing. When you first enter the V function, sector 360
is automatically accessed. DISKWIZ is designed to check
certain parameters of this sector and will stop to ask you
to verify that sector 360 is indeed the VTOC or to enter a
new VTOC sector number prior to proceeding if it appears
that 360 is not the real VTOC. This is because it is
possible to change the VTOC sector in DOS, or other
parameters at 360 itself, which many commercial programs
which use DOS do.

Because this command is sequential and does not check to
see if you are trying to lock or unlock already used or
unused sectors in the range specified, you can end up with
the wrong numbers for the 'free sectors' message. To
avoid this, map or otherwize check the disk to make sure
the range you give is not overlapping already used or
unused sectors. In case of accidental overlaps, map the
disk, count the uninverted codes, convert to hex (low
byte, high byte) and use the ALTER command on sector 360,
bytes 4 and 5, to correct the free sectors message.

W>WRITE SECTOR

The W>WRITE command allows you to write the sector
presently displayed to the disk. Enter "W", return. An
extra security prompt must then be answered, and you may
also choose to write to drive#2 at this time.

The A command when exited comes directly to this
function. Of course, answer negatively with RETURN, if
you are not yet ready to write to disk. If you intend to
return to this command later with the alterations you have
just made, do not make a call from the disk before you
have returned to this command for a write. Otherwise, you
will have to begin the changes from scratch.

After a write, the screen will refresh with what was just
written and return an updated status code. If the code is
not one, see your BASIC or other manuals for error code
conversion.

X>DISASSEMBLE

This command will allow you to disassemble m/l code at the
sector you are presently viewing.

Enter X, RETURN. The prompt will ask you for a hex or
decimal starting address. You may or may not be able to
determine the load address. Begin hex entries with a "$".
The default address for input errors or by pressing RETURN
only is $0000. You will then be asked for an offset in
decimal or hex. If you want to start from the first byte
of the sector, input a zero. If you want to skip over a
binary header, enter a six. Or if the first few bytes
appear to be throwing off the disassembly, enter whatever

number you like. The beginning row numbers in conjunction
with the hex column numbers will allow you to quickly
enter hex offsets. You will then be asked if you wish to
ignore the last three bytes of the sector, which you
should answer with a "Y" if you know that the sector is
DOS/OSA+ format. DISKWIZ loads two sectors at a time for
this function in order to continue into the next sector if
a command near the end of the sector crosses into the next.
At the end of each screen of disassembly a prompt asks if
you want to dump to a printer.   Hit RETURN if not and the
function will continue. Repeat to the end of the sector.
You will then be asked if you want to continue into the
next sector. RETURN if not.   If a "Y" was entered at
this time, the disassembly would continue into the next
sector using the appropriate offset and address from where
the previous sector ended.
You may also enter a plus or minus sign at the dump prompt
to page the screen forward or back one byte at a time
instead of a whole screen forward.   This is extremely
useful in correcting the first command byte of the
disassembly.
Be aware that the next sector may not necessarily be the
actual next sector if a load was actually taking place.

Y>SPEEDCHECK

Enter "Y", RETURN. A prompt will ask if you want to test
drive#2. If not, hit   RETURN.   The test will begin at
this time and will update every five seconds or so.   If
you inadvertently enter a "Y" when no drive#2 exists, hit
any key or the SRESET key to abort.   The speed in rpm
should be 288 plus or minus 3 rpm; for PERCOM drives, 295
to 301 rpm. The test will discontinue when any key is
pressed.
To adjust your 810 drive use a pen knife to remove the
four half inch diameter cover circles of the top side of
the drive. Then loosen the four phillips head screws in
the holes which were under the covers.   Remove the top
half of the drive housing.
Looking down into the drive you will see a circular
potentiometer at the left rear of older 810 drives or a
small rectangular blue/green potentiometer will a very
small adjustment screw near the center of the rear board
on newer 810's.   It's the only one there.   Using a small
screwdriver turn the adjustment wheel and simultaneously
watch the screen display. Make minor adjustments and then
wait for the display to update. Take care not to drop a
metal screwdriver or accidently short any bare circuitry
in the process.   Replace the cover of the drive
immediately when through to avoid troubles later.   See the
DUPLICATION segment following for more information.
For PERCOM drives, the adjuster is a long blue/green
potentiometer on a narrow circuit board on the rear of the
bare drive itself after removing the tan cover.   Remove
the glue drop before turning the adjustment screw.

Z>SEARCH

The SEARCH command will scan the entire disk for a string
that you define.
The string may be defined as a character string or hex/dec
bytes entered one at a time. Follow the prompts. When a
match is found, the normal screen display will reappear
and an arrow will point to the first byte of the match in
the hex field. You will then be asked if you wish to
continue the search for the same match. If not , hit
RETURN. If no match was found, the normal screen will
display the last sector searched at. If a match was
found and then no match after a continuation of the
search, the last matched sector will be displayed until
you press RETURN, at which time the screen will update to
the last sector accessed in the search. (end of range)
As with most other commands you can enter a range to
search. This function will also search across sector
boundaries into the next sector if the end of a sector
appears to be the beginning of your string. Also you may
choose to ignore link pointers if you are searching though
DOS type sectors by entering a "Y" when prompted.

+,-,SECTOR # ENTRY

Sector numbers can be entered in decimal, hex if preceded
by a "$", or you may page forward or back by entering a
plus or minus sign. This applies to calling up a sector#
at the main prompt. See the other commands for specific
hex or dec entry.

FIXING DUPLICATE FILENAMES

Use the A>ALTER command to change one of the filenames to
deleted by making its flag an inverted heart temporarily
or altering the name slightly in the directory entry.
Then exit DISKWIZ and use a disk with DOS to delete or
rename the files. You can also make your changes within
DISKWIZ by changing names or moving all zero sectors to
the sectors you want to delete and then using the VTOC
function to free up the sectors in the bit map.

RECOVERING DELETED FILES

If you have just deleted a file and then realized that you
should not have, you may be able to recover using DISKWIZ.
If the recovery attempt is made before DOS has written
anything else to your disk, your job will be quite simple:
Find the deleted file in the directory. Change the flag
byte to in use or locked. Use the trace command to locate
all of the sectors of the file and use the VTOC function
to lock these sectors again in the bit map.

If you have written more to the disk since the deletion,
you may or may not be able to recover. This depends on
the number of free sectors prior to the deleted file being
sufficient to have held all of the last write without
getting into the area of your old file. Use the mapping,
trace, and VTOC functions to inspect for the damage.
Also, the filename of the deleted file you want may have
been overwritten by the last file. In that case make a
new entry for the recovered file over another deleted name
or at the first all zero entry into the directory. If
part of the file has destroyed you may be able to save
part by changing links or adding dummy sectors at the
beginning. Study basic and binary files to see how they
are formed. If you have to make a new entry into the
directory make sure to adjust file name codes in the
sectors accordingly.

CHANGING BINARY HEADERS

DOS2/OSA+ binary headers can be altered to have the file
load at a different address. (See a DOS 1 manual for DOS
1 files)
There six bytes associated with these files. The first
two are "FF,FF" the next four designate the loading
addresses in memory, from then to, in the typical low byte
first order. Your DOS manual covers this subject fairly
well.
You can also add or delete bytes to short sectors or whole
sectors with different loading addresses if you wish.
When adding bytes or new short sectors, remember to change
the "used" byte, $7F(127), to reflect the new quantity.
If sectors are added before the first, start with a new
load address. If at the end, change the last sector of
the old file to point to your new sector, and put in
another new header to avoid changing a previous short
sector. Study your DOS manual and other binary files
using this program.

USING NON-FORMATTABLE DISKS

A non-formattable disk may have but one permanent
physically bad sector in order for DOS to refuse to
continue the entire formatting process. The disk will
have zeros written to all the good sectors. All you need
to do is locate the bad sectors using the map function.
Jot down the bad sector numbers or use the printout
functions. Duplicate sectors 1-3, and sectors 360 through
368 from a good blank formatted disk. Then use the VTOC
function to lock out the bad sectors from use by DOS.
If the disk will be used for non-DOS purposes, just make
sure that the unuseable sector or or sectors will not be
written to.

USING THE BACKSIDE

You can now use the backside of all of your disks in full
confidence, thereby doubling your storage capacity. All
you need to do is cut a new write protect notch just
opposite from the one that is already there. Use a second
disk flipped over to mark the location for the new notch.
Cut out this area with an Exacto type knife or use a
single hole paper punch. Use care not to distort the disk
itself. There is no disk material in the area of the
cover that you will remove.
It has been stated that there may be up to a 10% error
factor on the back side of disks. If it formats, no
problem. If it does'nt, see the previous section on
unformattable disks.
This will not work on PERCOM drives because they use the
small timing hole near the center of the disk. It is not
a good idea to make another one in the same fashion as
above for the notch since disk damage is more likely.

DISK DUPLICATION

Unlike DOS, disks are copied on a sector by sector basis
without regard to format, etc. Most disks are directly
copyable. Some disks will contain deliberately made "bad"
sectors (as opposed to physically bad disks). Some of
these bad sectors are readable, some are not. When
initially copying bad sectors, this program will copy all
unreadable bad sectors as a good all zero sectors. It
will copy readable bad sectors as good sectors with the
data it read from them. Many copy-protected disks try to
read a bad sector as part of their load routine. If they
read a good sector when there should be a bad one, they
won't run.
A simple bad sector may be made by drastically adjusting
your drive speed out of whack. Experiment with speeds.
220 rpm is a good place to start. Sectors nearer 720 may
require a little faster speed. Sectors nearer to 1, a
little slower. Then use the **W**, **R**, or **D** commands to
write to the sector or sectors that should be "bad".
Re-adjust your speed to normal and check to make sure the
damage has been done. The more sophisticated "bad
sectoring" technique is done with special machinery. In
order to duplicate these disks to run, you will have to
disassemble and patch the code. This subject is beyond
the realm of this manual. You are free to experiment. Do
not overlook the value of duplicating a disk even if you
can't get the dup to run. The vast majority of glitches
showing up on your copy-protected disks will be on good
sectors. You will be able to refresh the bombed good
sectors with the backup.
Some newer ATARI 810 drives will require a 2000 ohm pot or

resistor placed temporarily across R105 on the rear board of the drive in order to slow the drive sufficiently. This may be accomplished with temporary clips and a resistor straddling R105. No speed adjustment is necessary in this case as the speed will change automatically to around 214, and will go back to its previous setting after the resistor is removed. PERCOM drives will not write bad sectors when slowed down but will when speeded up. The only problem is that the adjacent sector on the disk (not necessarily consecutive) will also be made bad. In some cases this will be ok; in others, not ok.

Remember that it is only legal to duplicate disks that you legally own for backup purposes only.

MISC

The OS (operating system) boot number is located at sector 1, the second byte ($01), of all disks whether DOS/OSA+ or not. This byte tells the OS how many consecutive, non-linked sectors to boot before turning control over to the loaded program. This can be of the range 1-256 (0=256). Usually on a DOS/OSA+ disk this number will be 3. After the program or DOS takes over from the OS, more sectors may be loaded as part of the boot process.

Tokenized BASIC (SAVED) files sometimes look the the same as m/l binary files. The difference can be determined on the first sector of the file. If the files first two bytes are 'FF FF' then the file is machine language. If the first bytes are '00 00 00 01' then the file is SAVED BASIC. If you can read it, the file is LISTED, or source code or text data.

APPENDIX A: REFERENCES

DE RE ATARI
From APX (APX-90008)

OPERATING SYSTEM AND HARDWARE USERS MANUAL (TECHNICAL REF
NOTES)
ATARI, INC., SUNNYVALE, CA. 94086
Part# C016555

ATARI DISK OPERATING SYSTEM II REF MANUAL
Part# C016347

ASSEMBLER EDITOR USER'S MANUAL
ATARI Part# C014189-03

INSIDE ATARI DOS
by BILL WILKINSON
COMPUTE! BOOKS
POB 5406, GREENSBORO, N.C. 27403

"DISK DATA STRUCTURES: AN INTERACTIVE TUTORIAL"
by DAVID YOUNG
COMPUTE!
pp156-161, MARCH 1982

APPENDIX B: NORMAL DISPLAYS

```
00>|03  20  56  E4  AD  00  01  C9  ↵  Vd■♥ ▮I
08>|59  D0  38  A9  00  85  08  A2  YP8)♥▮ ▮
10>|20  A9  0C  9D  42  03  20  56  ▮ ▮8↵  V
18>|E4  A5  0C  8D  9C  17  A5  0D  d%■ ■↑%■
20>|8D  9D  17  A9  40  85  0C  A9  ■▮ ▮@■T▮
28>|15  85  0D  A9  2F  A2  10  9D  ■T▮/■+■@
30>|44  03  A9  18  9D  45  03  A0  D↵▮▮↵■E↵
38>|00  8C  9E  15  88  8C  9D  15  ♥■◄■■ ▮■
40>|20  A4  15  60  45  3A  9B  44  ■▄◆E ▮■D
48>|31  3A  44  55  50  2E  53  59  1:DUP.SY
50>|53  9B  45  52  52  4F  52  2D  S■ERROR-
58>|53  41  56  49  4E  47  20  55  SAVING U
60>|53  45  52  20  4D  45  4D  4F  SER MEMO
68>|52  59  20  4F  4E  20  44  49  RY ON DI
70>|53  4B  9B  54  59  50  45  20  SK■TYPE
78>|59  20  54  4F  20  00  26  7D  Y TO ♥&⌐
```

```
SECTOR:37     STATUS:1     FILE:A
USED:125      NEXT:38
ENTER SECTOR# OR COMMAND?G
■
```

location column **hex field** **char field**

```
00>.03  20  56  E4  AD  00  01  C9  . Vd-..I
08>.59  D0  38  A9  00  85  08  A2  YP8)..."
10>.20  A9  0C  9D  42  03  20  56  )..B. V
18>.E4  A5  0C  8D  9C  17  A5  0D  d%....%.
20>.8D  9D  17  A9  40  85  0C  A9  ...)@..)
28>.15  85  0D  A9  2F  A2  10  9D  ...)/"..
30>.44  03  A9  18  9D  45  03  A0  D.)..E.
38>.00  8C  9E  15  88  8C  9D  15  ........
40>.20  A4  15  60  45  3A  9B  44  $. 'E:.D
48>.31  3A  44  55  50  2E  53  59  1:DUP.SY
50>.53  9B  45  52  52  4F  52  2D  S.ERROR-
58>.53  41  56  49  4E  47  20  55  SAVING U
60>.53  45  52  20  4D  45  4D  4F  SER MEMO
68>.52  59  20  4F  4E  20  44  49  RY ON DI
70>.53  4B  9B  54  59  50  45  20  SK.TYPE
78>.59  20  54  4F  20  00  26  7D  Y TO .&.
```

This display (and all others) are from the same disk as map.

```
SECTOR:37     STATUS:1     FILE:A
USED:125      NEXT:38
ENTER SECTOR# OR COMMAND?P
```

APPENDIX C: THE DIRECTORY

```
FILE   QTY      START       NAME          FLAG
------------------------------------------------
A      39       4           DOS      SYS   B
B      42       43          DUP      SYS   B
C      14       85          ENTRY1PT1      B
D      18       99          ENTRY2         B
E      18       117         ENTRY2    1    B
F      14       135         ENTRY1PT2      B
G      18       149         ENTRY3         B
H      28       167         ENTRY4   1UP   B
```

```
SECTOR 361   DIRECTORY
FLAGS:
  B  =  ($42) DOS.2 NORMAL
  ♥  =  ($80) NEVER USED
  ✖  =  ($80) DELETED
  b  =  ($62) LOCKED FILE
  C,c  =  ($43,$63) OPEN FILE


ENTER SECTOR# OR COMMAND?G
■
```

```
00> |42  27  00  04  00  44  4F  53  B' ♥ H DOS ⟩ file entry# 0
08> |20  20  20  20  20  53  59  53          SYS ⟩ (coded "A")
10> |42  2A  00  2B  00  44  55  50  B♥♥+♥DUP
18> |20  20  20  20  20  53  59  53          SYS
20> |42  0E  00  55  00  45  4E  54  B_♥U♥ENF
28> |52  59  31  50  54  31  20  20  RY1PT1
30> |42  12  00  63  00  45  4E  54  B—♥c♥ENT
38> |52  59  32  20  20  20  20  20  RY2
40> |42  12  00  75  00  45  4E  54  B—♥u♥ENT
48> |52  59  32  20  20  31  20  20  RY2   1
50> |42  0E  00  87  00  45  4E  54  B_♥■♥ENT
58> |52  59  31  50  54  32  20  20  RY1PT2
60> |42  12  00  95  00  45  4E  54  B—♥■♥ENT
68> |52  59  33  20  20  20  20  20  RY3
70> |42  1C  00  A7  00  45  4E  54  B↑♥■♥ENT
78> |52  59  34  20  20  31  55  50  RY4   1UP

SECTOR:361    STATUS:1    FILE:DIREC
USED:40    NEXT:362
ENTER SECTOR# OR COMMAND?G
■
```

APPENDIX D: THE VTOC (bit map sector)

first byte
of bit map

available sectors
(707)

free sectors
(284)

```
00>|02 C3 02 1C 01 00 00 00   C |+ ♥♥♥
08>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
10>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
18>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
20>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
28>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
30>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
38>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
40>|07 FF FF FF FF FF FF FF   \]]]]]]]
48>|FF FF FF FF FF FF FF FF   []]]]]]]
50>|FF FF FF FF FF FF FF FF   []]]]]]]
58>|FD FF FF FF FF FF FF FF   []]]]]]]
60>|FF FF FF FF 00 00 00 00   ]]]]♥♥♥♥
68>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
70>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
78>|00 00 00 00 00 00 00 00   ♥♥♥♥♥♥♥♥
```

```
SECTOR:360   STATUS:1   FILE:DIREC
USED:128   NEXT:361
ENTER SECTOR# OR COMMAND?G
■
```

last byte
of bit map

diskwiz

APPENDIX E: DISK MAP AND FILE TRACE

```
  1  AEYAAAAAAAAAAAAAAAAAAAAAAAAAAAA   30        boot sectors
 31  AAAAAAAAAAABBBBBBBBBBBBBBBBBBBB   60
 61  BBBBBBBBBBBBBBBBBBBBBBBBBCCCCCC   90        in use
 91  CCCCCCCCDDDDDDDDDDDDDDDDDDDEEEE  120
121  EEEEEEEEEEEEEEFFFFFFFFFFFFFFGG   150        VTOC
151  GGGGGGGGGGGGGGGGGHHHHHHHHHHHHH  180
181  HHHHHHHHHHHIIIIIIIIIIIIIIIIIII  210
211  IIIIIIIJJJJJJJJJJJJJJJJJJJJJJJ  240        directory
241  JJJJKKKKKKKKKKKKKKKKKKKKKKKKKK  270
271  KKLLLLLLLLLLLLLLLLLLLLLLLLLLLL  300
301  LLLLLLLLLOPPPPPQQQRRRRRRRRRR    330        not in use (free)
331  RRRRRRRRRRRRRRRRRRRRRRRRRRRRRA  360        (deleted)
361  MITA....RRRRRRRRRRRRRRRRRRRRR   390
391  PPPPPPPPPPPPPPPPPPPPPPPPPPPPWX  420        all zeros
421  WXPPPPPPPPPPPPPPPQQQQQQQQQQQQQ  450
451  QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ  480        "bad sector"
481  YYYYYYYYYYYYYYYYYYYYYYYYYYYYYY  510        (or error)
511  YYYYYYYYYYYYYYYYYYYYYYYYYYYYYY  540
541  YYYYYYYYY.....................  570        non-DOS
571  ............................o  600
601  .............................  630
631  .............................  660
661  .............................  690
691  .......7kI1AOTIA.............  720
```

```
  1  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥   30
 31  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥   60
 61  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥   90
 91  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  120
121  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  150
151  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  180
181  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  210
211  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  240
241  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  270
271  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  300
301  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥RRRRRRRR  330
331  RRRRRRRRRRRRRRRRRRRRRRRRRRRRR♥  360
361  ♥♥♥♥♥♥♥♥RRRRRRRRRRRRRRRRRRRRRR  390
391  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  420
421  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  450
451  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  480
481  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  510
511  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  540
541  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  570
571  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  600
601  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  630
631  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  660
661  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  690
691  ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥  720
```

Trace of file "R" on same disk as shown above.

diskwiz

APPENDIX F: SPECIAL MESSAGE PRINTER

```
THIS IS AN EXAMPLE OF INSERTING AN
ADDITIONAL MESSAGE ON THE SCREEN▬♥▣
PRIOR TO USING EITHER PRINTOUT ▶♥◤◢◣
COMMAND, P> OR G>. WRITE IN WHAT YOU
WANT AND THEN ENTER A "P" OR "G" AND
RETURN ON THE FIRST AVAILABLE BLANK▶
LINE PAST THE PROMPT→↓10 9D ▬▜▷/▣♦▣
30>|44 03 A9 18 9D 45↓03 A0 D◢▶┙◖E◢▮
38>|00 8C 9E 15 88 8C↓9D 15 ♥◢▄◢/◢▆
40>|20 A4 15 60 45 3A↓9B 44 �◢♦E:◧▛D
48>|31 3A 44 55 50 2E↓53 59 1:DUP.SY
50>|53 9B 45 52 52 4F↓52 2D S◧▛ERROR-
58>|53 41 56 49 4E 47↓20 55 SAVING U
60>|53 45 52 20 4D 45↓4D 4F SER MEMO
68>|52 59 20 4F 4E 20↓44 49 RY ON DI
70>|53 4B 9B 54 59 50↓45 20 SK◧▛TYPE
78>|59 20 54 4F 20 00↓26 7D Y TO ♥&▆
                         ↓
SECTOR:37    STATUS:1  ↓ FILE:A
USED:125     NEXT:38    →→→→→↓
ENTER SECTOR# OR COMMAND? ↓
↓←←←←←←←←←←←←←←←←←←←←←←←←←←←
G
■
```

```
THIS IS AN EXAMPLE OF INSERTING AN
ADDITIONAL MESSAGE ON THE SCREEN-..I
PRIOR TO USING EITHER PRINTOUT )..."
COMMAND, P> OR G>. WRITE IN WHAT YOU
WANT AND THEN ENTER A "P" OR "G" AND
RETURN ON THE FIRST AVAILABLE BLANK)
LINE PAST THE PROMPT..10 9D ...)/"..
30>.44 03 A9 18 9D 45.03 A0 D.)..E.
38>.00 8C 9E 15 88 8C.9D 15 ........
40>.20 A4 15 60 45 3A.9B 44 $. 'E:.D
48>.31 3A 44 55 50 2E.53 59 1:DUP.SY
50>.53 9B 45 52 52 4F.52 2D S.ERROR-
58>.53 41 56 49 4E 47.20 55 SAVING U
60>.53 45 52 20 4D 45.4D 4F SER MEMO
68>.52 59 20 4F 4E 20.44 49 RY ON DI
70>.53 4B 9B 54 59 50.45 20 SK.TYPE
78>.59 20 54 4F 20 00.26 7D Y TO .&.
                         .
SECTOR:37    STATUS:1 . FILE:A
USED:125     NEXT:38   ......
ENTER SECTOR# OR COMMAND? .
..........................
P
```

NOTES:

HEX	DEC	CHR	KEY
00	000	♥	c,
01	001	⊦	cA
02	002	|	cB
03	003	⌐	cC
04	004	⊣	cD
05	005	¬	cE
06	006	/	cF
07	007	\	cG
08	008	◢	cH
09	009	▪	cI
0A	010	◣	cJ
0B	011	▪	cK
0C	012	▪	cL
0D	013	─	cM
0E	014	_	cN
0F	015	▪	cO

HEX	DEC	CHR	KEY
10	016	↟	cP
11	017	r	cQ
12	018	─	cR
13	019	+	cS
14	020	●	cT
15	021	⊣	cU
16	022	|	cV
17	023	⊤	cW
18	024	⊥	cX
19	025	▮	cY
1A	026	∟	cZ
1B	027	⌐	esc
1C	028	↑	ec↑
1D	029	↓	ec↓
1E	030	←	ec←
1F	031	→	ec→

HEX	DEC	CHR	KEY
20	032		spc
21	033	!	!
22	034	"	"
23	035	#	#
24	036	$	$
25	037	%	%
26	038	&	&
27	039	'	'
28	040	(	(
29	041	)	)
2A	042	*	*
2B	043	+	+
2C	044	,	,
2D	045	-	-
2E	046	.	.
2F	047	/	/

HEX	DEC	CHR	KEY
30	048	0	0
31	049	1	1
32	050	2	2
33	051	3	3
34	052	4	4
35	053	5	5
36	054	6	6
37	055	7	7
38	056	8	8
39	057	9	9
3A	058	:	:
3B	059	;	;
3C	060	<	<
3D	061	=	=
3E	062	>	>
3F	063	?	?

HEX	DEC	CHR	KEY
40	064	@	@
41	065	A	A
42	066	B	B
43	067	C	C
44	068	D	D
45	069	E	E
46	070	F	F
47	071	G	G
48	072	H	H
49	073	I	I
4A	074	J	J
4B	075	K	K
4C	076	L	L
4D	077	M	M
4E	078	N	N
4F	079	O	O

HEX	DEC	CHR	KEY
50	080	P	P
51	081	Q	Q
52	082	R	R
53	083	S	S
54	084	T	T
55	085	U	U
56	086	V	V
57	087	W	W
58	088	X	X
59	089	Y	Y
5A	090	Z	Z
5B	091	[	[
5C	092	\	\
5D	093	]	]
5E	094	^	^
5F	095	_	_

HEX	DEC	CHR	KEY
60	096	♦	c.
61	097	a	a
62	098	b	b
63	099	c	c
64	100	d	d
65	101	e	e
66	102	f	f
67	103	g	g
68	104	h	h
69	105	i	i
6A	106	j	j
6B	107	k	k
6C	108	l	l
6D	109	m	m
6E	110	n	n
6F	111	o	o

HEX	DEC	CHR	KEY
70	112	p	p
71	113	q	q
72	114	r	r
73	115	s	s
74	116	t	t
75	117	u	u
76	118	v	v
77	119	w	w
78	120	x	x
79	121	y	y
7A	122	z	z
7B	123	♠	cj
7C	124	|	s|
7D	125	▙	esCLR
7E	126	◀	eDEL
7F	127	▶	eTAB

HEX	DEC	CHR	KEY
80	128	▮	c,
81	129	▮	cA
82	130	▮	cB
83	131	▮	cC
84	132	▮	cD
85	133	▮	cE
86	134	▮	cF
87	135	▮	cG
88	136	▮	cH
89	137	▮	cI
8A	138	▮	cJ
8B	139	▮	cK
8C	140	▮	cL
8D	141	▮	cM
8E	142	▮	cN
8F	143	▮	cO

HEX	DEC	CHR	KEY
90	144	▮	cP
91	145	▮	cQ
92	146	▮	cR
93	147	▮	cS
94	148	▮	cT
95	149	▮	cU
96	150	▮	cV
97	151	▮	cW
98	152	▮	cX
99	153	▮	cY
9A	154	▮	cZ
9B	155	invt	RET
9C	156	▮	esDEL
9D	157	▮	esINS
9E	158	▮	ecTAB
9F	159	▮	esTAB

HEX	DEC	CHR	KEY
A0	160	▮	SPC
A1	161	▮	!
A2	162	▮	"
A3	163	▮	#
A4	164	▮	$
A5	165	▮	%
A6	166	▮	&
A7	167	▮	'
A8	168	▮	(
A9	169	▮	)
AA	170	▮	*
AB	171	▮	+
AC	172	▮	,
AD	173	▮	-
AE	174	▮	.
AF	175	▮	/

HEX	DEC	CHR	KEY
B0	176	▮	0
B1	177	▮	1
B2	178	▮	2
B3	179	▮	3
B4	180	▮	4
B5	181	▮	5
B6	182	▮	6
B7	183	▮	7
B8	184	▮	8
B9	185	▮	9
BA	186	▮	:
BB	187	▮	;
BC	188	▮	<
BD	189	▮	=
BE	190	▮	>
BF	191	▮	?

HEX	DEC	CHR	KEY
C0	192	▮	@
C1	193	▮	A
C2	194	▮	B
C3	195	▮	C
C4	196	▮	D
C5	197	▮	E
C6	198	▮	F
C7	199	▮	G
C8	200	▮	H
C9	201	▮	I
CA	202	▮	J
CB	203	▮	K
CC	204	▮	L
CD	205	▮	M
CE	206	▮	N
CF	207	▮	O

HEX	DEC	CHR	KEY
D0	208	▮	P
D1	209	▮	Q
D2	210	▮	R
D3	211	▮	S
D4	212	▮	T
D5	213	▮	U
D6	214	▮	V
D7	215	▮	W
D8	216	▮	X
D9	217	▮	Y
DA	218	▮	Z
DB	219	▮	[
DC	220	▮	\
DD	221	▮	]
DE	222	▮	^
DF	223	▮	_

HEX	DEC	CHR	KEY
E0	224	▮	c.
E1	225	▮	a
E2	226	▮	b
E3	227	▮	c
E4	228	▮	d
E5	229	▮	e
E6	230	▮	f
E7	231	▮	g
E8	232	▮	h
E9	233	▮	i
EA	234	▮	j
EB	235	▮	k
EC	236	▮	l
ED	237	▮	m
EE	238	▮	n
EF	239	▮	o

HEX	DEC	CHR	KEY
F0	240	▮	p
F1	241	▮	q
F2	242	▮	r
F3	243	▮	s
F4	244	▮	t
F5	245	▮	u
F6	246	▮	v
F7	247	▮	w
F8	248	▮	x
F9	249	▮	y
FA	250	▮	z
FB	251	▮	c;
FC	252	▮	s|
FD	253	▮	ecCLR
FE	254	▮	ecDEL
FF	255	▮	ecINS

DISKWIZ-II

ENTER: SECTOR(DEC/HEX)/LETTER/OR +,−

A>ALTER DATA	**R**>RENUMBER/MOVE
D>DUP SECTORS	**S**>SECTOR LINK
E>ENTRY:DIR. FILE	**T**>TRACE FILE
F>FORMAT:DIRECTORY	**U**>UTILITY MENU
G>GRAFIX PRINT	**V**>VTOC/BIT MAP
H>HEX-DEC-ASC CONV	**W**>WRITE SECTOR
L>LIST COMMANDS	**X**>DISASSEMBLE
M>MAP DISK	**Y**>SPEEDCHECK
P>PRINTOUT	**Z**>SEARCH

RETURN TO REDISPLAY DATA

SRESET TO ABORT

WARRANTY FORM

Please fill out the information below,
remove this page and mail to:
        Allen Macroware
        PO BOX 2205
        Redondo Bch., CA 90278

Program: DISKWIZ-II Serial#_____

NAME: _____
ADDRESS: _____
             _____
CITY: _____
STATE: _____ ZIP_____
PHONE (OPT): ____ _____
PURCHASED FROM: _____
                _____
                _____
                _____
DATE PURCHASED: _____

PRINTER USED: _____
DRIVE BRAND/MODEL: _____
INTERFACE TYPE: _____
ATARI COMPUTER MODEL: _____
DID YOU SEE DISKWIZ ADVERTISED?_____
IF SO, WHERE?_____
             COMPUTE?_____
             ANALOG?_____
             ANTIC?_____
             ROM?_____
             OTHER?_____

WHAT PARTICULAR TYPE OF PROGRAM WOULD
YOU LIKE TO SEE US WRITE?
_____
_____
_____
_____