TI - D - BITS

PHILADELPHIA AREA USERS GROUP NEWSLETTER
COVERING THE TI99/4A
AND MYARC 9640 COMPUTERS

SEPTEMBER 1990

Volume 10 Number 7

WE ARE HERE

And We are Looking for the rest

of you in this NEW FALL SEASON !!

THE PHILADELPHIA AREA TI-99/4A USERS' GROUP (SEP '90)

The Philadelphia Area TI-99/4A Users' Group meets twice a month. On the first Saturday of any given month, we meet at the Bucks County Youth Development Center, (YDC, which is next to Neshaminy Mall), Administration Building, beginning at 10:00 am. On the third Saturday of each month, we meet at LaSalle University, 20th Olney, in room H-329 located in the Science Building. Membership to The Philadelphia Area TI-99/4A Users' Group is available to all. We invite anyone that is interested in the TI-99/4A to visit us. Stop in and see what is available to you for your TI and how membership can benefit you!

Current executive board consists of:

PRESIDENT...........	Allan Silversteen.	215-885-7910
VICE PRESIDENT......	Eric Bray.........	215-848-5515
SECRETARY...........	Mark Wannop.......	609-365-1776
TREASURER...........	Don Arsenault.....	215-368-0446

Committees consists of:

TI-d-BITS	Ralph Field.......	215-362-2534
	Don Arsenault.....	215-368-0446
	OPEN	
	Rice Hall	
LIBRARY	Ted Chemey........	215-752-1458
	Rich Mascara	215-441-4060
MEMBERSHIP ...	OPEN	
ASSISTANT TREASURER.	Frank Passini	
EDUCATION	Barry Traver	
	Frank Passini	
	Ted Chemey	
	Tim Coyne	
	Carlo Angelico	
EQUIPMENT	Rice Hall	
PROGRAM	Dr. Eric Bray	

REMEMBER to be considerate when calling any of the above people. Limit your calls to the early evening hours. (6pm to 9pm)

The opinions expressed herein are those of the individual authors are not necessarily those of the Philadelphia Area TI-99/4A Users' Group or its officers. Nor is the Philadelphia Area TI-99/4A Users' Group or any of its officers responsible for any damage, inconvenience, or loss which may result as a consequence of the use of any written material herein.

TI-d-Bits is published monthly by the Philadelphia Area TI-99/4A Users' Group, c/o Don Arsenault, 1290 Buttonwood Dr., Lansdale, PA 19446. All material herein may be reprinted freely by other non-profit User Groups, (unless otherwise stated), as long as proper credit is given to both source and author. Contributions are encouraged, but no payment is made. Editorial, advertising, and classified, copy MUST be in by the LAST day of the previous month. You can either mail your copy to: TI-d-Bits, c/o The Philadelphia Area TI-99/4A Users' Group, c/o Don Arsenault, 1290 Buttonwood Dr., Lansdale, PA 19446 or send it via modem by contacting Don Arsenault at (215)-368-0446. If your piece contains any diagrams, charts, or code, send a paper copy AT FINAL PUBLICATION SIZE.

Classified ads are printed in blocks. A block consists of 3 lines, 55 characters wide, or any increment of 3 lines. Classified advertising is accepted from members at NO CHARGE for a one block ad, per issue. Additional ads from members may be placed at cost of $1.00 per block. Non-members may place classified ads at a cost of $2.00 per block. All advertisements MUST be paid for in advance.

Commercial advertising is accepted for publication at the following rates:

Quarter page	$ 5.00
Half page	$ 8.00
Full page	$15.00

Commercial advertisements will be placed in the next available issue. All advertisements MUST be paid for in advance.

The editor of TI-d-Bits or the executive board of The Philadelphia area TI-99/4a Users' Group reserve the right to reject any material submitted for publication for any reasons.

The Philadelphia Area TI-99/4A Users' Group's program library is available to all active members at NO CHARGE for copying to your disk. A charge of $2.00 per disk is made for club supplied disks for members. Non members may obtain copies of the library for a fee of $5.00 per disk. A catalog of the library's contents is given to all new members upon request and updates will appear in this publication from time to time. To obtain material from the library, contact the librarian for the best procedure to obtain your requests.

****

ASSEMBLER PROGRAM FILES

By Norm Sellers

****

On the TI99 computers, when a disk directory says "Program File", you may have a BASIC program or you may have an assembler program. Today, I would like to discuss Assembler Program Files, their characteristics, and even give you a tool to be able to edit these files (instead of re-assembling and re-linking the program) for minor changes.

An assembler program is often on disk as several files with the last character of the name of each file incremented by one. For example, TI Writter program is written to disk as two files, EDITA1 and EDITA2. The reason that multiple files are used is that when the file is read into memory, it is first read directly into a very limited (approximately 2K) size of VDP memory. From here, it is written to console memory according to the information in the first six bytes of the program file (I will refer to this as Section 1 of the file). Table I describes this information more clearly.

TABLE I

WORD	DESCRIPTION
1	End-of-load flag: =0 means this is the last file to load. not=0 means there is atleast 1 more file to load.
2	Length of program data/code to be loaded.
3	Address in console memory to write the program to for execution.

The entry point (first statement to execute when running the program) of the standard multiple file program is defined to be the first byte after section I of the first file loaded.

Assembler program files are memory image with all addresses resolved before load time, therefore there are no tags, as in the tagged object files created by assemblers.

This really makes editing the program file simpler than editing object code which contains absolute and relocatable addresses. However, the first problem in editing a program file *s to find a program that edits 'program files'. The only way I have found to do this is using a Sector Editor program.

This adds a complication to editing a program file because you can not load the whole thing in memory at one time but only >100 bytes of the program. You need to know what sector to read for editing and where to go in the sector.

When you start to edit the program file at a particular spot in your assembly program, you suddenly find that the memory image code does not have much of an indication of where a sector is loaded in memory and the code in the sector looks different from the hex assembler generated code in the program listing mainly because the addresses are all absolute in the sector.

To get around these problems, I have used the facts that I know:

A) Where the first byte of the program file is loaded (from section 1) and,

B) the fact that every byte of data in the program file is loaded in memory byte by byte.

Therefore, if I want to edit a statement in the program that is loaded >20 bytes from the beginning of the program, I would simply go to displacement >26 in the first program sector (remember section 1 took 6 bytes and is not loaded in memory with the program).

Of course the corrections usually are at more like >900 bytes into the program (not in the first sector). It is therefore necessary to use a program like DSKU to see what sectors are in the file. Also, it is necessary to follow the sectors carefully when the file has been fragmented.

Well, after doing these calculations by hand several times, I decided to let the computer do the work so I wrote a small BASIC program to do the work. The following program does these calculations:

```
100 ! SAVE DSK2.PGMPATCH
110 ! By Norm Sellers
120 ! Sept. 1989
130 !
140 J=1
150 INPUT "PRINT?(Y OR N)":ANS$
160 IF ANS$<>"Y" THEN 180
170 OPEN #1:"RS232.BA=4800.DA=8"
180 HEX$-"0123456789ABCDEF"
190 INPUT "Enter file name=":FN$
200 IF ANS$<>"Y" THEN 220
210 PRINT #1:"FOR FILE="&FN$
220 INPUT "Enter LOAD adr=>":STRT$
230 CALL HX2DC(STRT$,ADR,HEX$)
240 PADR=ADR
250 FRST=1
260 DIM SF(10),SL(10),PADRS(10),PHX$(10)
270 FOR I=1 TO 10
280 INPUT "Enter FIRST SEG NUM=>":SGF$
290 IF SGF$="QUIT" THEN 350
300 IF SGF$="" THEN 350
310 INPUT "Enter LAST  SEG NUM->":SGL$
320 CALL HX2DC(SGF$,SF(I),HEX$)
330 CALL HX2DC(SGL$,SL(I),HEX$)
340 NEXT I
350 NSS=I-1
360 PRINT "ENTER LOW TO HIGH ORDER:"
370 PRINT "PATCH ADDRS THEN ENT 'QUIT'"
380 FOR I=1 TO 10
390 INPUT STR$(I)&" HEX ADDR=:":PHX$(I)
400 IF PHX$(I)="QUIT" THEN 440
410 IF PHX$(I)="" THEN 440
420 CALL HX2DC(PHX$(I),PADRS(I),HEX$)
430 NEXT I
440 NPS=I-1
450 J=1
460 FOR I=1 TO NSS
470 FOR S=SF(I)TO SL(I)
480 IF FRST=1 THEN PADR=PADR-6 ELSE PADR=PADR+256
490 FRST=0
500 IF PADRS(J)-PADR>256 THEN 590
510 DISP=PADRS(J)-PADR
520 CALL DC2HX(DISP,DISP$,HEX$)
530 CALL DC2HX(S,S$,HEX$)
540 CALL DC2HX(PADR,PAD$,HEX$)
550 PRINT PHX$(J)&" "&S$&"  "&DISP$
```

```
560 IF ANS$="Y" THEN PRINT #1:PHX$(J)&" "&$&" "&DISP$
570 J=J+1
580 IF J>NPS THEN 620 ELSE 500
590 NEXT S
600 NEXT I
610 GOTO 280
620 IF ANS$="Y" THEN CLOSE #1
630 STOP
640 !
650 SUB HX2DC(A$,A,H$)
660 A=POS(H$,SEG$(A$,1,1),1)-1
670 L=LEN(A$)
680 FOR I=2 TO L
690 A=A\16+POS(H$.SEG$(A$.I.1).1)-1
700 NEXT I
710 SUBEND
720 !
730 SUB DC2HX(AA,A$,H$)
740 A=AA
750 A$=""
760 Q=INT(A/16)
770 HD=A-Q\16
780 A$=SEG$(H$,HD+1,1)&A$
790 IF Q=0 THEN SUBEXIT
800 A=Q
810 GOTO 760
820 SUBEND
```

Before using the program, you should enter your printer name in statement 170 and save the program again.

The program first asks if you wish to print the results. Answer, 'Y' or 'N'. Then you are requested for the program file name to edit. This is only needed and used for your future reference. It is printed with the results.

The program then loops asking for the first and last sectors in the file (for upto 10 portions of a fragmented file). After you have entered the last sector in the file, just hit <ENTER> to get out of the current input loop.

Next you are requested for the addresses where corrections are needed in your program file that you are debugging. These addresses must be in increasing address order. Again, to get out of the input loop, just hit <ENTER>.

The program responds with the list of addresses you are editing with their corresponding locations in the program file given as 'sector number' and 'displacement' for each address. This list is on the screen and maybe the printer if you said 'Y' to 'PRINT (Y OR N).

Any program changes are possible, as long as the replacement code or changes do not require any more memory than previously used by the program. Remember this is all memory image with all addresses explicitly given in changed statements (relative statements like JMP still behave the same as shown in the program listing though).

I have been using this method of correcting assembly programs for several years and find it mush easier and faster (and saves a lot of paper by listing the assemblies). Infact, it is nice to make a correction and immediately run it (just like in BASIC).

Try it next time you have a program correction to make. For the first time or so, I would suggest backing up your program to edit incase Murphy is too near. Enjoy.

****

CHEZ GENEVE
By Chip Chapin
Fm MUG Newsletter
University Park,Pa

****

The other day I ran across this listing on GEnie and thought I would share it with you. It came from Radio Free Milwaukee, WI (414) 352-6176, and had apparently been left on the TI-ECHO. Scott Stasiowski is the accredited auther. The paragraphs following the listing were also on GEnie, but I am not sure that Scott is the author.

MDOS TIDBITS

COMMANDS	WHAT IT PERFORMS	COMMENTS
DIR `D	Shows DISPLAY type filenames.	Catalogs your disk and looks for only selected filetypes you specify with the DIR ` command.
DIR `DV	Shows DISPLAY VARIABLE type filenames.	
DIR `DF	Shows DISPLAY FIXED type filenames.	
DIR `IV	Shows INTERNAL VARIABLE type fileames.	
DIR `IF	Shows INTERNAL FIXED type filenames.	
DIR `P	Shows PROGRAM type filenames.	
COPY `D	Copies all DISPLAY types.	Using the ` command with the COPY command will copy All specified file types on yoru disk to be copied.
COPY `DV	Copies all DISPLAY VARIABLE types.	
COPY `DF	Copies all DISPLAY FIXED files.	
COPY `IV	Copies all DISPLAY VARIABLE files.	
COPY `IV	Copies all INTERNAL FIXED files.	
COPY `IF	Copies all PROGRAM files.	
DEL `D	Deletes all DISPLAY types.	Using the ` command with the DEL command will then Delete All specified filetype to be deleted!
DEL `DV	Deletes all DISPLAY VARIABLE types.	
DEL `DF	Deletes all DISPLAY FIXED files.	
DEL `IF	Deletes all INTERNAL VARIABLE files.	
DEL `IV	Deletes all INTERNAL FIXED files.	
DEL `P	Deletes all PROGRAM files.	
ATTRIB +P `D	Protects all DISPLAY types.	Using ATTRIB +P command with the command will cause all specified file type to become Protected.
ATTRIB +P `DV	Protects all DISPLAY VARIABLE types.	
ATTRIB +P `DF	Protects all DISPLAY FIXED files.	
ATTRIB +P `IV	Protects all INTERNAL VARIABLE files.	
ATTRIB +P `IF	Protects all INTERNAL FIXED files.	
ATTRIB +P `P	Protects all PROGRAM files.	
ATTRIB -P `D	Un-Protects all DISPLAY types.	Using ATTRIB -P command with the command will cause all specified file types to become Un-Protected.
ATTRIB -P `DV	Un-Protects all DISPLAY VARIABLE type	
ATTRIB -P `DF	Un-Protects all DISPLAY FIXED files.	
ATTRIB -P `IV	Un-Protects all INTERNAL VARIABLE	
ATTRIB -P `IF	Un-Protects all INTERNAL FIXED files.	
ATTRIB -P `P	Un-Protects all PROGRAM files.	

```
560 IF ANS$="Y" THEN PRINT #1:PHX$(J)&" "&$&" "&DISP$
570 J=J+1
580 IF J>NPS THEN 620 ELSE 500
590 NEXT S
600 NEXT I
610 GOTO 280
620 IF ANS$="Y" THEN CLOSE #1
630 STOP
640 !
650 SUB HX2DC(A$,A,H$)
660 A=POS(H$,SEG$(A$,1,1),1)-1
670 L=LEN(A$)
680 FOR I=2 TO L
690 A=A\16+POS(H$.SEG$(A$.I.1).1)-1
700 NEXT I
710 SUBEND
720 !
730 SUB DC2HX(AA,A$,H$)
740 A=AA
750 A$=""
760 Q=INT(A/16)
770 HD=A-Q\16
780 A$=SEG$(H$,HD+1,1)&A$
790 IF Q=0 THEN SUBEXIT
800 A=Q
810 GOTO 760
820 SUBEND
```

Before using the program, you should enter your printer name in statement 170 and save the program again.

The program first asks if you wish to print the results. Answer, 'Y' or 'N'. Then you are requested for the program file name to edit. This is only needed and used for your future reference. It is printed with the results.

The program then loops asking for the first and last sectors in the file (for upto 10 portions of a fragmented file). After you have entered the last sector in the file, just hit <ENTER> to get out of the current input loop.

Next you are requested for the addresses where corrections are needed in your program file that you are debugging. These addresses must be in increasing address order. Again, to get out of the input loop, just hit <ENTER>.

The program responds with the list of addresses you are editing with their corresponding locations in the program file given as 'sector number' and 'displacement' for each address. This list is on the screen and maybe the printer if you said 'Y' to 'PRINT (Y OR N).

Any program changes are possible, as long as the replacement code or changes do not require any more memory than previously used by the program. Remember this is all memory image with all addresses explicitly given in changed statements (relative statements like JMP still behave the same as shown in the program listing though).

I have been using this method of correcting assembly programs for several years and find it mush easier and faster (and saves a lot of paper by listing the assemblies). Infact, it is nice to make a correction and immediately run it (just like in BASIC).

Try it next time you have a program correction to make. For the first time or so, I would suggest backing up your program to edit incase Murphy is too near. Enjoy.

SCROLLING THE COMMAND STACK

In the Geneve MDOS manual on P.34 they mention Scrolling the Command Stack. Well, the Geneve can do this but with two changes I have noticed. The FCTN UP and DOWN arrows are used. NOT the F1 as the manual states!

SCROLLING THE SCREEN

In the Geneve manual on P.35 they state that F3 allows you to see again the Text etc that has gone off your screen. Well again the Geneve 9640 can do this but you MUST use the FCTN 6 to Scroll UP the screen and then you could also use the FCTN 4 key to Scroll DOWN the screen.

NOTE: While in this SCROLL MODE you can also use your UP and DOWN arrows for minor adjustments.

FILENAMES

Have you ever tried to rename a file from MY/FILE to some other name while in MDOS? Well MDOS gets upset and doesn't think it is a valid filename. The / is reserved for special MDOS commands. So a way around it is to type:

REN MY"?"FILE MYFILE

Did you notice the " " that I used? Well that is a li'l trick that Don Walden from the Milawaukee Area User Group showed me when I last had him over for a visit some ancient time ago.

SCREEN COLORS

Looking through the manual I did not find a reference to any color setup, but there is one! You can use the following commands either in your BATCH files or in your AUTOEXEC files or simply just by typing them in at your A: or whatever drive you are at:

```
MODE Fn    Foreground / Where n equals any number from 1-16
MODE Bn    Background / Where n equals any number from 1-16
```

==

one of my songs into it.

```
100 CALL CLEAR :: CALL SCREE
N(2):: FOR X=4 TO 12 :: CALL
 COLOR(X,16,2):: NEXT X
110 CALL CHARPAT(39,A$,44,B$
,38,C$):: CALL CHAR(61,A$,96
,B$,64,C$)
120 DISPLAY AT(2,7):"SEDUCTI
ON WALTZ":"":"Words @ music
  Program by":"by Jim Peters
on Lucie Dorais"
130 FOR D=1 TO 700 :: NEXT D
140 CALL CLEAR :: CALL START
(95,23,95,199)
150 LB=123 :: LC=131 :: LD=1
47 :: LE=165 :: LF=175 :: LG
=196 :: LA=220 :: B=494 :: C
=523 :: D=587 :: E=659 :: F=
698 :: G=784 :: A=880
160 HB=988 :: HC=1047 :: RV=
5 :: CV=9
170 DIM M$(16):: FOR J=1 TO
16 :: READ M$(J):: NEXT J
180 GOTO 190 :: A$,TIME,X,Y
:: CALL DELSPRITE :: !@P-
190 FOR TIME=1 TO 4
200 Y=Y+1 :: DISPLAY AT(1,1)
:M$(Y):: CALL BAR(RV,CV,-RV,
-CV,G,C,C,LC,LE,LG)
210 CALL BAR(0,CV,0,-CV,C,D,
D,LC,LE,LG)
220 CALL BAR(-RV,CV,RV,-CV,C
,E,E,LC,LE,LG)
230 CALL BAR(-RV,0,RV,0,E,E,
E,LC,LE,LG)
240 Y=Y+1 :: DISPLAY AT(1,1)
:M$(Y):: CALL BAR(-RV,-CV,RV
,CV,G,C,C,LC,LE,LG)
250 CALL BAR(0,-CV,0,CV,C,D,
D,LC,LE,LG)
260 CALL BAR(RV,-CV,-RV,CV,C
,E,E,LC,LE,LG)
270 CALL BAR(RV,0,-RV,0,E,E,
E,LC,LE,LG)
280 Y=Y+1 :: DISPLAY AT(1,1)
:M$(Y):: CALL BAR(-RV,CV,RV,
-CV,G,A,A,LF,LA,LC)
290 CALL BAR(0,CV,0,-CV,A,F,
G,LF,LA,LC)
300 CALL BAR(RV,CV,-RV,-CV,A
,G,A,LF,LC,LA)
310 CALL BAR(RV,0,-RV,0,G,E,
E,LC,LE,LG)
320 Y=Y+1 :: DISPLAY AT(1,1)
:M$(Y):: CALL BAR(RV,-CV,-RV
,CV,G,G,D,LG,LB,LD)
330 CALL BAR(0,-CV,0,CV,F,E,
B,LG,LB,LD)
340 CALL BAR(-RV,-CV,RV,CV,D
,C,C,LC,LE,LG)
350 CALL BAR(-RV,0,RV,0,C,C,
C,LC,LE,LG)
360 CALL PATTERN(#1,132,#2,1
28):: CALL MOTION(#1,0,0,#2,
0,0)
370 NEXT TIME
380 CALL BAR(RV,0,RV,0,G,G,D
,LG,LB,LD)
390 CALL BAR(RV,0,RV,0,F,E,H
C,LG,LB,LD)
400 CALL BAR(RV,0,RV,0,HB,HC
,HC,LC,LE,LG)
410 CALL DELSPRITE(ALL)
420 CALL BAR(RV,0,RV,0,HC,HC
,HC,LC,LE,LG)
430 DISPLAY AT(24,11):"AGAIN
? Y" :: ACCEPT AT(24,18)SIZE
(-1)VALIDATE("YN"):A$
440 IF A$="N" THEN END ELSE
DISPLAY AT(24,11):"" :: Y=0
:: CALL START(95,23,95,199):
: GOTO 190
450 !@P+
460 DATA Dance with me by th
e light  of the moon,And swa
y to the throbbing   guitars
,The love that we feel is a
 love that is real
470 DATA And it=s witnessed
by only  the stars,Is it wro
ng to love and be  loved?,Is
 it wrong to kiss and     ca
ress?,Is it wrong to rest in
 the  arms'
480 DATA Of the one you love
 the      best?,What=s the di
fference what  others may sa
y?,Does it matter who is to
    blame?
490 DATA When the wind of sp
ring=s inyour hair,And the m
oon makes your      hearts be
at the same
500 DATA "Come with me' let
us run'    let us go",In the
darkness and no one  will se
e,Come with me to a place th
atI know
510 DATA On this wild night
of love  stay with me
520 !@P+
530 SUB START(R1,C1,R2,C2)
540 CALL CHAR(128,"070F0A0B0
B0307070F0F1F1F3F3F3F021C1C0
8FC1CFC1C1C9C88C8C8E8E8E818"
)
550 CALL CHAR(132,"3838103F3
83F383839111313171717180E0F05
0D0D0C0E0E0F0F0F8F8FCFC40")
560 CALL CHAR(136,"030301030
303030307070F0F1F1F1F0280800
080808080800C0C0E0E0F0F0F080"
)
570 CALL MAGNIFY(4):: CALL S
PRITE(#1,128,8,R1,C1,#2,132,
10,R2,C2):: SUBEND
580 SUB BAR(R1,C1,R2,C2,T1,T
2,T3,B1,B2,B3)
590 P2=128-4*(P1=128):: P1=1
28-4*(P2=128)
600 CALL PATTERN(#1,P1,#2,P2
):: CALL MOTION(#1,R1,C1,#2,
R2,C2):: FOR T=1 TO 4 :: CAL
L SOUND(-999,T1,5,T1*1.01,5,
B1,10):: NEXT T
610 FOR T=1 TO 6 :: CALL SOU
ND(-999,T2,5,T2*1.01,5,B2,10
):: NEXT T :: CALL PATTERN(#
1,136,#2,136)
620 FOR T=1 TO 4 :: CALL SOU
ND(-999,T3,5,T3*1.01,5,B3,10
):: NEXT T :: P1=P2 :: SUBEN
D
```

If you're watching your diet -

```
100 DISPLAY AT(1,1)ERASE ALL
:"NUTRITION LABEL INTERPRETE
R":"      by Jim Peterson":""
:"  To help you understand t
hemandatory FDA nutrition
  labels on food packages."
110 DISPLAY AT(8,1):"Calorie
s per serving?" :: ACCEPT AT
(8,23)VALIDATE(NUMERIC)BEEP:
C
120 DISPLAY AT(10,1):"Grams
of fat?" :: ACCEPT AT(10,15)
VALIDATE(NUMERIC)BEEP:F
130 DISPLAY AT(12,1):"Grams
of sucrose & other":"sugars?
" :: ACCEPT AT(13,9)VALIDATE
(NUMERIC)BEEP:S
140 DISPLAY AT(15,1):"Grams
of starch and other":"carboh
ydrates?" :: ACCEPT AT(16,16
)VALIDATE(NUMERIC)BEEP:K
150 F=INT(F*9/C*100+.5):: S=
INT(S*4/C*100+.5):: K=INT(K*
4/C*100+.5)
160 DISPLAY AT(18,1):STR$(F)
&"% of calories from fat":ST
R$(S)&"% of calories from su
gars":STR$(K)&"% of calories
 from":"starches and carbohy
drates"
170 DISPLAY AT(22,1):STR$(10
0-F-S-K)&"% of calories from
 dietary fiber."
180 GOTO 110
```

I came across this beauty of a routine in the Summer 1989 newsletter of the Lehigh 99'er Computer Group. Assemble it with only the R option, as TRACK/O.

```
***************************
*    BOOT TRACKING        *
*   Adrian Robinson       *
*      May 1989           *
* CALL LINK("TRACK",A$)   *
***************************

       DEF   TRACK
STRASG EQU   >2010
VMBR   EQU   >202C
MYWS   BSS   32
H30    BYTE  >30    >3011
       BYTE  11
DSK    BSS   11
       EVEN
TRACK  LWPI  MYWS
       LI    R0,>3FF5
       LI    R1,DSK
       LI    R2,11
       BLWP  @VMBR
       AB    @H30,@DSK
       CLR   R0
       LI    R1,1
       LI    R2,DSK-1
       BLWP  @STRASG
       LWPI  >83E0
       CLR   @>837C
       B     @>70
       END
```

When you load a program, the DOS saves its drive number at >3FF5 and the filename in the next 10 bytes. By recovering this drive number, you can write your program to open and read files from whatever drive the program may be loaded from.

You can load this routine by a CALL LOAD, but that must be preceded by a CALL INIT which will wipe out the record in >3FF5, so you will need to start the program with this line -

```
100 CALL INIT :: DISPLAY AT(
12,1)ERASE ALL:"DRIVE NO.?"
:: ACCEPT AT(12,12):D$ :: CA
LL LOAD("DSK"&D$&".TRACK/O")
:: CALL LINK("TRACK",A$)
```

Then, when you want to open a file,

```
OPEN #1:"DSK"&SEG$(A$,1,1)&"
."&(the filename)
```

However, if you imbed this routine in your program with Todd Kaplan's invaluable ALSAVE routine, you can omit the LOAD routine!

And to fill the column -

```
100 CALL CLEAR :: FOR J=1 TO
 12 :: CALL COLOR(J,16,2)::
NEXT J
110 CALL SCREEN(2):: DISPLAY
 AT(3,5):"SNOWFALL ON GANYME
DE":;:;:" THE SNOWFLAKES ON
THE THIRD":;:;:"MOON OF JUPI
TER ARE LARGE"
120 DISPLAY AT(12,1):"AND IN
 MANY COLORS, BUT LIKE":;:;:
"THOSE OF EARTH THEY ARE":;:
;:"ALWAYS SYMMETRICAL AND NO
"
130 DISPLAY AT(21,1):"TWO AR
E EVER THE SAME."
140 C=3 :: Y=1 :: CALL MAGNI
FY(3):: RANDOMIZE :: FOR CH=
40 TO 120 STEP 4
150 FOR J=1 TO 8 :: FOR K=1
TO 8 :: Z=INT(12*RND+2):: X=
INT(Z*RND+1):: X$=X$&STR$(AB
S(X=Z)):: Y$=STR$(ABS(X=Z))&
Y$ :: NEXT K
160 CALL BIN_HEX(X$,H$):: A$
=A$&H$ :: B$=H$&B$
170 Y$=SEG$(Y$,2,7)&"0" :: C
ALL BIN_HEX(Y$,H$):: C$=C$&H
$ :: D$=H$&D$ :: X$,Y$="" ::
 NEXT J
180 CALL CHAR(CH,A$&SEG$(B$,
3,14)&"00"&C$&SEG$(D$,3,14))
:: CALL CLEAR :: CALL SPRITE
(#Y,CH,C,150,150,15*RND-15*R
ND,15*RND-15*RND)
190 A$,B$,C$,D$="" :: C=C+1+
(C=16)*14 :: Y=Y+1 :: NEXT C
H
200 GOTO 200
210 SUB BIN_HEX(B$,H$):: HX$
="0123456789ABCDEF" :: BN$="
0000X0001X0010X0011X0100X010
1X0110X0111X1000X1001X1010X1
011X1100X1101X1110X1111"
220 L=LEN(B$):: IF L/4<>INT(
L/4)THEN B$="0"&B$ :: GOTO 2
20
230 FOR J=L-3 TO 1 STEP -4 :
: X$=SEG$(B$,J,4)
240 X=(POS(BN$,X$,1)-1)/5 ::
 T$=SEG$(HX$,X+1,1)&T$ :: NE
XT J :: H$=T$ :: T$="" :: SU
BEND
```

*** EQUIPTMENT FOR SALE ****

TI99/4A Converted with RAVE Keyboard System	$200.00
TI99/4A Computer	$ 50.00
PE BOX (With the Following:)	$100.00
Flex Cable Interface	$ 50.00
32K Memory	$ 99.00
RS232 Interface Card (TI)	$ 50.00
TRIPTECH Card (Speach,Clock,64K Buffer)	$150.00
9900 Corcomp Disk Controller	$170.00
TI Disk Drive	$ 50.00
DS/DD External Disk Drive	$ 99.00
TI Disk Controller Card	$ 50.00
TI Modules: (including Manuals)	
Personal Real Estate	$ 5.00
Household Budget Mgr.	$ 5.00
TI Logo II	$ 10.00
Mini Memory	$ 10.00
Disk Mgr.	$ 5.00
TI ExBasic	$ 10.00
Home Finance Decisions	$ 5.00
Terminal Emulator II	$ 10.00
Securities Analysis	$ 5.00
Editor/Assembler	$ 10.00
Tax Investment/Rec. Keeping	$ 5.00
Personal Rec. Keeping	$ 5.00
Report Generator	$ 5.00
Microsoft/Multiplan	$ 10.00
Navarone:	
Data Base Mgr.	$ 30.00
3 Moduler Selector (Widget)	$ 15.00
TI Disk Software:	
TI Word Processor	$ 10.00
Cash Mgr.	$ 5.00
Checkbook Mgr.	$ 5.00
Data Base Mgr.	$ 5.00
TI Logo Sampler	$ 5.00
Computer Music Box	$ 5.00
Tech. ExBasic	$ 5.00
	$ 1250.00

Plus 100's of Books, Mags, Articles, Extra Cables, etc.

The above prices are for individual pieces and each large piece is negotiable on an individual basis.

TAKE THE ENTIRE SYSTEM AND EXTRA COMPUTER FOR . . . . $ 350.00

Ernest D. Brown 15 Charles St. Auburn NY 13021 * Phone 315 253 6947

(or contact me Ralph Field, your TI-D-Bits Editor
This person's son is my neighbor.