HOCUS

Home Computer Users Spotlight

A monthly publication of the Milwaukee Area 99/4 Users Group

OCTOBER, 1984

H O C U S F O C U S

THERE'S STILL PLENTY AVAILABLE OUT THERE

It has now been one year since TI has orphaned the 99/4A. Everyone was sure that their computing days were in peril. But quite remarkably, the 99/4A has held on. Most of the popular peripherals are still quite easy to locate. Perhaps the most significant hardware developments are in the area of disk drives. Last year you couldn't touch double sided drives for under $240. The other day I saw an ad in BYTE Magazine advertising them for an unbelievable price of $175. TI developed software is also still around. While the availability of some titles is questionable, the recent price cuts are certainly not. TI has dropped the prices charged to dealers significantly. Two weeks ago, I drove to Lake Geneva and purchased 10 TI cartridges for under $65 at Prange's. Price cuts are also in evidence in the latest catalogs and listings from TRITON and the BACH Company. Additionally, the old standby sources, Toys-R-Us, Comp-U-Serv, and Competition still have TI items in stock. Third party hardware and software vendors seem to be everywhere as well. The way I see it, our "orphan" has been quite nicely "adopted" by an entire world of "parents" and appears to be headed towards a healthy life. -TK

\- - - - - - - - -

NEWSLETTER CHANGES

The November issue of the Milwaukee Area 99/4A Users Group newletter will be the premier of a new concept and publication of its kind. The co-editors of the newletter will be Peter Radike and Gary Pichler, with Contributing Editor George Kasica.

Without revealing too much detail on the content, format, or appearance of the November Issue, members are being advised here that many improvements and special features will be included for their enjoyment.

Since the newsletter is designed for the benefit of the Milwaukee Area 99/4A User Group, several alternatives are being considered to assure that only paying members and officers obtain it.

ACTIVE PARTICIPATION of non-editor members will be encouraged to see that it becomes a newsletter FOR the members and BY the members (Not just 1 or 2). -PR

WHAT THE HECK DO I DO WITH MY EDITOR/ASSEMBLER?

By Peter Radike

After all the hoopla over TI getting out of the business of making and supplying a Home Computer, I decided to rush to get an Editor/Assembler software package (as well as other module software that was to become almost impossible to get anywhere). I, like many others, planned to "learn assembly on the TI some day", and figured that I should get the "E/A" software now.

Well, it's been 6 months....and you know what? I still don't know didly about assembly -BUT- I have found a way that I can use my E/A almost daily and not have to labor through reems of reference materials and manuals.

Presto. This article was written entirely with Editor / Assembler's Editor feature. While it is by no means a fullblown multi-feature Word Processing program, you do have many of the "convenience" abilities of TI Writer. Full screen controls for editing and text manipulation are constantly in the reach of any typist. AND IT MEANS YOU WILL ONLY NEED ABOUT 15 MINUTES OF READING SIMPLE INSTRUCTIONS TO LEARN IT.

What you need to get started is the E/A package (which is made up of a module, two diskettes, and the E/A manual) and the TI expansion box with disk-drive and 32k card. Now, just do the following simple steps:

1. Turn on your TI and insert the E/A module.
2. Insert the E/A diskette with the EDIT1 program into Disk Drive 1.
3. Press 1 for the E/A option on the TI menu screen and again Press 1 to load the editor part of the diskette into memory.
4. Press FCTN and 9 to default the screen and Press 2 to enter the "edit mode".

There are various options you have while you are in this mode... so I suggest you read the part of your E/A manual that deals with how to use the Editor (pages 21-32). You will have the ability to save your work to disk for future reference or you can print your text on your printer.

I hope many more of you take the opportunity to utilize this new capability with Editor/Assembler. You will begin to get some of your money's worth out of this piece of software as soon as you take the time (15 minutes) to learn it's full potential as a moderately, respectable Word Processor... Unless of course you plan to use it for what it really was designed for - Assembly Language programming!! HAPPY COMPUTING.....

COMPUTER QUIZ

SOFTWARE
1. ANGORA SWEATER
2. FOAM RUBBER FORKS

DISK DRIVE
1. CAMPAIGN TO COLLECT OLD RECORDS
2. SAUCER SHAPED PARKING PLACE

FLOW CHART
1. MAP OF RIVERS IN THE AREA
2. GRAPH THAT FELL INTO THE SOUP

INVENTOR OF THE COMPUTER
1. MR. CHIPS
2. BILL COSBY

WROTE TI COMPUTER MANUALS
1. E.T.
2. MARQUIS DE SADE

MICROCHIPS
1. EATEN WITH MICRO DIP
2. WHAT A HERD OF MICROS DROP ON THE PRAIRIE

FLOPPY DISK
1. RECORD ALBUM THEY CAN'T SELL
2. PAINFUL LOWER BACK PROBLEM
3. RUBBER FRISBEE

FORTRAN
1. BETWEEN THREE TRAN AND FIVE TRAN
2. HOW COMPUTERS GET EXCITED PRIOR TO INTERFACE

PASCAL
1. SOUTHERN FRENCH CITY
2. FOOT FUNGUS
3. LEAFY VEGETABLE

RELIABLE COMPUTER ORGANIZATION
1. IBM
2. CIA
3. PLO

ANSWERS NEXT MONTH.....

H O C U S

HOME COMPUTER USERS SPOTLIGHT

HOCUS IS PUBLISHED MONTHLY BY THE MILWAUKEE AREA 99/4A USERS GROUP, [illegible] N. 71ST STREET, WAUWATOSA, WI 53213. THE MILWAUKEE AREA 99/4A USERS GROUP IS AN ASSOCIATION OF INDIVIDUALS WITH A COMMON INTEREST IN USING AND PROGRAMMING TEXAS INSTRUMENTS 99/4A HOME COMPUTERS. THE MILWAUKEE AREA 99/4A USERS GROUP IS NOT AFFILIATED WITH TEXAS INSTRUMENTS INC., NOR ANY OTHER COMMERCIAL ORGANIZATIONS.

HOCUS IS PUBLISHED FOR THE MEMBERS OF THE MILWAUKEE AREA 99/4A USERS GROUP AND IS COMPOSED OF ARTICLES WRITTEN AND DONATED BY USER GROUP MEMBERS. OPINIONS EXPRESSED BY THE AUTHORS DO NOT NECESSARILY REPRESENT THOSE OF HOCUS. ANY ARTICLE APPEARING IN THIS PUBLICATION MAY BE REPRODUCED PROVIDING CREDIT IS GIVEN TO THE AUTHOR AND TO HOCUS.

MEMBERSHIP INFORMATION

MEMBERSHIP IS OPEN TO INDIVIDUALS AND FAMILIES WHO ARE INTERESTED IN USING AND PROGRAMMING THE TEXAS INSTRUMENTS 99/4A HOME COMPUTER. THE MEMBERSHIP INCLUDES ACCESS TO BOTH THIS NEWSLETTER AND TO THE USER GROUP LIBRARY. ANNUAL DUES ARE: INDIVIDUAL, $8.00; FAMILIES, $12.00. TO JOIN, SEE THE TREASURER AT ANY OF OUR MONTHLY MEETINGS.

MEETING INFORMATION

THE MILWAUKEE AREA 99/4A USERS GROUP MEETS ON THE LAST SATURDAY OF EACH MONTH IN THE LOWER LEVEL OF WAUWATOSA SAVINGS & LOAN AT 7500 W. STATE STREET IN WAUWATOSA. MEETING TIME IS 1:00 TO 4:00 P.M..

SPECIAL NOTE: DUE TO A SCHEDULING CONFLICT DURING 1984, THE DECEMBER MEETING WILL BE HELD ON THE THIRD SATURDAY OF THE MONTH (DEC 15TH), AT OUR NORMAL TIME AND PLACE.

USERS GROUP OFFICERS:

PRESIDENT

JIM VINCENT
782-9353

VICE-PRESIDENT

MILTON GIESSEN
251-2864

TREASURER

JEROME TRINKL
[illegible]-0170

CORRESPONDING SECRETARY

GENE HITZ
453-0499

RECORDING SECRETARY

JUDY BROWN
1-677-2894

USER GROUP LIBRARY:

LIBRARIAN

STEVE SANDERS
546-1821

NEWS LETTER COMMITTEE:

MANAGING EDITORS

TOM KRUSE
475-1159

MIKE MILDE
784-0479

CONTRIBUTING EDITORS

JIM KUNDINGER
541-1999

STEVE TJENSVOLD
962-4924

TERMINAL EMULATOR II TUTORIAL
Reprinted from Data Stream
(Janesville Users Group)

This tutorial was originally written to be placed on the Source.

Here are some answers to your questions about downloading text from BBS's to your TI-99/4A. Also, I'll pass along a short Extended Basic utility program called CHANGE, which can be used with the TI-WRITER word processor and Terminal Emulator II (we call it TEII for short) to upload mesages writtem with TI-WRITER to a TIBBS (tm) System, as well as Source's or POST.

First, TEII. It allows you to copy a "page" of data to another port or device using the Control-2 command. Remember that a page can either be from 34 to 40 character wide, or 80 characters wide, although you can see only a 40-character "window" in that mode. If you are copying to a disk drive, you have to leave TEII with the Control-Zero command, or your file is lost. (More experienced TI users will know there are a couple of ways to retrieve this data, but that's for another story.)

TEII files are saved on disk in Display/Fixed 80 or Display Variable 80 format. These files can be directly loaded into TI-WRITER (on which this is being written) and edited from there. TI-WRITER saves files in Display/Variable 80 format. And, TEII's "auto-logon" function accepts files in Display/Variable 80 format. Put these together, and you have the combination that will allow you to upload a text file using the two programs. Page 28 of the TEII manual has an odd explanation about hex bytes and so forth. Forget that. Here's what you need in a TEII logon file: --In front of each line of data. you need to stick the character "1". --Then, if you want to delay sending the next line, put in a line that is made of the character "2", and another character that tells TEII how long the delay should be. (This next part gets a little complicated.) The module figures the delay by dividing by 60 the ASCII number of the character that follows the "2". For example, a small "x" has an ASCII number of 120, so TEII will wait for two seconds before it feeds the next line of text. Now, here's the listing of the CHANGE program, written in Extended Basic, to be used with TI-WRITER:

```
100 CALL CLEAR
120 INPUT "NAME OF FILE TO BE CONVERTED: ":F$
140 OPEN #2:"DSK1."&F$&"C"
160 OPEN #1:"DSK1."&F$,INPUT ,DISPLAY ,VARIABLE 80
180 PRINT "CONVERTING LINE NUMBER ":
200 X=1
220 LINPUT #1:A$
240 IF EOF(1)=1 THEN 380
260 A$="1"&A$&CHR$(13)
280 PRINT X
300 PRINT #2:A$
320 PRINT #2:"2"&CHR$(120)
340 X=X+1
360 GOTO 220
380 CLOSE #1
400 CLOSE #2
420 PRINT "CONVERSION COMPLETE"
440 END
```

Here's the explanation of the CHANGE listing: LINE 120 prompts the user to type in the name of the file (minus the "DSK1." the computer needs.) The space is because of the 28-column TI format. LINE 140 opens the duplicate file, with the first filename, plus a "C". LINE 160 opens the original text file. LINE 180 lets you know the file was found. LINE 200 sets up a variable for the line count to be shown on the screen. LINE 220 gets each line of the text file. "LINPUT" exists only in Extended Basic. Standard "INPUT" ends the line when a comma is found. LINE 240 tells the program when to end. LINE 260 changes the text line so it can be read by TEII, adding the "1" at the beginning. At the end, "CHR$(13)" is a carriage return character, needed so the remote system will know the end of the line has been reached. LINE 280 prints the number of the line being processed on the screen. LINE 300 stores the converted text line in the duplicate disk file. LINE 320 places the characters "2x" in the duplicate file. "2" tells TEII to wait, and "x" tells it to wait for two seconds. LINE 340 increments the line count, and LINE 360 sends the program to the line to check if the end of the original file has been reached. LINE 380 and LINE 400 close the disk files. LINE 420 tells the user the duplicate file has been finished. Here's the way it all goes together: If you see a file from another system you wish to copy, use the Control-2 command. Remember to use Control-Zero to close the disk file. (A=Abort, any other key to cont.)

Load the file into TI-WRITER and edit as you wish. IMPORTANT: Set the right margin to 76. If you have a line that's longer than 77 characters, the CHANGE program will make the line too long for TEII to use. Also, keep your file name short, no more than four characters. When you finish editing, run CHANGE and make the duplicate file. If the original file is named TEST, the duplicate file will be named TESTC. When you're ready to feed the file down to another system, go to the auto-login screen of TEII, and type in the duplicate file name. The file will scroll out, line by line, with two seconds between lines. NOTE: if you use the "enter" key in TI-WRITER to end a line, add this line to CHANGE: 250 IF SEG$(A$,LEN(A$),1)=CHR$(13) THEN A$=SEG$(A$,1,LEN(A$)-1) This will lop the c/r off the end of the line. Some systems recognize two consecutive c/r's as a sign you've finished uploading your file.

```
HAR(R,R+4,33,26-R*2):: NEXT
R
150 FOR R=13 TO 24 :: CALL H
CHAR(R,29-R,34,(R-12)*2):: N
EXT R
180 FOR C=5 TO 16 :: CALL VC
HAR(C-4,C,35,34-C*2):: NEXT
C
210 FOR C=17 TO 28 :: CALL V
CHAR(29-C,C,36,C*2-33):: NEX
T C
225 FOR J=0 TO 7 :: A$(J+1),
B$(8-J)=SEG$("00000000000000
",1,2*J)&"FF" :: NEXT J
230 C$(1),D$(8)=RPT$("80",8)
:: C$(2),D$(7)=RPT$("40",8):
: C$(3),D$(6)=RPT$("20",8)::
 C$(4),D$(5)=RPT$("10",8)
240 C$(5),D$(4)=RPT$("08",8)
:: C$(6),D$(3)=RPT$("04",8):
: C$(7),D$(2)=RPT$("02",8)::
 C$(8),D$(1)=RPT$("81",8)
250 FOR C=2 TO 15 :: FOR J=1
 TO 8 :: CALL CHAR(33,A$(J),
34,B$(J),35,C$(J),36,D$(J)):
: NEXT J :: CALL SCREEN(C)::
 NEXT C :: GOTO 250
```

Next, I would like to share with you a gem of a "why didn't I think of that" routine which John Taylor sent me.

```
100 ! 28 COLUMN TEXT ROUTINE
IN EXTENDED BASIC (EASILY
CONVERTED TO BASIC) BY JULIE
PACK, B.U.G., P.O. BOX 1402
PALM BAY, FL 32906
110 ! ENHANCED BY JET
SHOALS 99'ERS, P.O. BOX 2928
MUSCLE SHOALS, AL 35662
120 CALL CHAR(64,"00282828")
130 ! PROGRAM TO COPY STARTS
      HERE
140 CALL CLEAR :: X=-1
150 RESTORE
160 IF X>=21 THEN X=1 :: CAL
L WAIT
170 READ MESS$
180 IF MESS$="P" THEN DISPLA
Y AT(X+2,1):Z$ :: X=X+4 :: Z
$="" :: GOTO 160
190 IF MESS$="ZZZ" THEN DISP
LAY AT(X+2,1):Z$ :: CALL WAI
T :: END
200 IF LEN(Z$)>0 THEN MESS$=
Z$&" "&MESS$
210 X=X+2
220 IF X>=21 THEN X=1 :: CAL
L WAIT
230 IF LEN(MESS$)<29 THEN DI
SPLAY AT(X,1):MESS$ :: Z$=""
 :: GOTO 160
240 FOR A=1 TO 29
250 I=POS(MESS$," ",A)
260 IF (I=0 OR I>29)AND A=1
THEN A,J=29 :: GOTO 290
270 IF I=0 OR I>29 THEN A=29
 :: GOTO 290
280 J,A=I
290 NEXT A
300 IF X>=21 THEN DISPLAY AT
(X,1):SEG$(MESS$,1,J-1):: X=
-1 :: CALL WAIT :: GOTO 320
310 DISPLAY AT(X,1):SEG$(MES
S$,1,J-1)
320 IF SEG$(MESS$,J,1)=" " T
HEN I=1 ELSE I=0
330 Z$=SEG$(MESS$,J+I,163)::
 MESS$=Z$ :: IF LEN(Z$)>28 T
HEN X=X+2 :: GOTO 240
340 GOTO 160
350 DATA "THIS SHORT ROUTINE
 WILL ENABLE YOU TO WRITE LO
NG TEXT MATERIAL IN YOUR DAT
A STATEMENTS SO YOU WON'T HA
VE TO WORRY ABOUT COUNTING"
360 DATA "THE LENGTH OF YOUR
 SENTENCES ALL THE TIME.  TH
IS ROUTINE WILL AUTOMATICALL
Y EDIT YOUR TEXT TO FIT A 28
 COLUMN SCREEN."
370 DATA "A SUGGESTION- IT I
S A GOOD IDEA TO PUT A QUOTE
 AT THE BEGINNING AND END OF
 THE DATA STATEMENTS SO YOU
WON'T HAVE TO WORRY ABOUT"
380 DATA "COMMAS LIKE THIS ,
,, AND THEY WILL REMAIN IN Y
OUR TEXT PROPERLY."
390 DATA "THIS ROUTINE WILL
ALSO CLEAR THE SCREEN (WHEN
FILLED) AND CONTINUE READING
 YOUR DATA AND DISPLAYING YO
UR TEXT ON THE NEXT SCREEN."
400 DATA P
410 DATA "  TO START A NEW P
ARAGRAPH ENTER THE LETTER @P
@ AS A SEPERATE DATA STATEME
NT, THEN INDENT YOUR TEXT ON
 YOUR NEXT NEXT DATA"
420 DATA "STATEMENT 2 OR 3 S
PACES (IF DESIRED).",P,"TO S
KIP LINES,",P,"JUST ENTER @P
@",P,"WHERE EVER YOU WANT TO
",P,"SKIP."
430 DATA P,"MAKE SURE THAT Y
OUR VERY LAST DATA STATEMENT
 IS @ZZZ@, AND JUST REPLACE
THESE DATA STATEMENTS WITH"
440 DATA "YOUR OWN.",P,"YOU'
LL ALSO FIND THIS ROUTINE IS
 MOST USEFUL WHEN CONCATENAT
ING STRINGS, E.G., @ELIZA@ T
YPE PROGRAMS-",P
450 DATA "AN EXAMPLE:",P,"A$
=@JACK AND JILL WENT UP@","B
$=@THE HILL TO FETCH A@","C$
=@PAIL OF WATER.@","D$=A$&B$
&C$&D$","PRINT D$",P
460 DATA "JACK AND JILL WENT
 UP THE HILL TO FETCH A PAIL
 OF WATER.",P,P,P,"HAPPY PRO
GRAMMING!"
470 DATA ZZZ
480 SUB WAIT
490 DISPLAY AT(24,8):"PRESS
ANY KEY"
500 CALL KEY(0,K,S):: IF S=0
 THEN 500 ELSE CALL CLEAR
510 SUBEND
```

Thank you, Julie and John. This is becoming one of the most useful routines on my utility disk. I was preparing a disk of PD programs for our UG library. Some of them needed extra instructions, so I typed them out on TI-Writer, so that people could run them off on their printer. Then I remembered that some folks don't have printers. So -

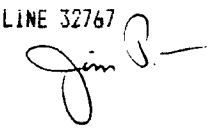
```
50 CALL CLEAR :: INPUT "FILE
NAME? DSK1."?:F$
60 DIM B$(150):: OPEN #1:"DS
K1."&F$,INPUT, DISPLAY ,VAR
IABLE 80
70 A=A+1 :: LINPUT #1:B$(A)
80 IF EOF(1)=1 THEN B$(A+1)=
"ZZZ" ELSE 70
```

and change line 170 to -

```
170 @=@+1 :: MESS$=B$(@)
```

And there you have a quickie program to check out those DIS/VAR 80 files that show up on your disks under filenames that you can't remember using.

MEMORY FULL IN LINE 32767



TIPS FROM THE TIGERCUB

#16

TIGERCUB SOFTWARE
156 Collingwood Ave.,
Columbus OH 43213

Distributed by Tigercub Software to TI-99/4A Users Groups for promotional purposes and in exchange for their newsletters. May be reprinted by non-profit Users' Groups, with credit to Tigercub Software.

These Tips are being mailed, together with my new catalog #5, to every Users Group that I know of. I hope that you will make both the Tips and the catalog available to your membership. I am sorry that I cannot take out paid ads in your newsletters, but to advertise in each one of them would cost me more than I have made in the past 6 months, and I would not get enough business to break even.

If you would like to continue receiving these Tips, put me on the mailing list for your newsletter, and give me some indication that my Tips are really reaching your members and not going into someone's private file. If I receive enough business from this mailing to pay for its cost, I will then continue to send you my Tips. If not, this will be the last issue of the Tips from the Tigercub.

Copies of my catalog are available for $1.00, which is deductable from your first order. I have over 130 absolutely original quality programs in Basic, many of them now also available in XBasic, on casette or disk for only $3.00 each plus $1.50 per order for casette, package and postage, or $3.00 for diskette, package and postage (higher overseas). I give one-day service, I give bonuses for repeat orders, I give bonus programs on diskette orders.

In addition, any User's Group member who mentions his/her users'group when sending me an order before 1 Jan. 1985 may deduct 10% from the cost of the programs.

Tips from the Tigercub #1 thru #14 are now available, with more added, as a diskfull of 50 programs, routines and files for only $15 postpaid.

I have also now completed my NUTS & BOLTS disk of 100 XBasic utility subprograms in MERGE format, ready to merge into your own programs, for just $19.95 postpaid.

In The last Tips, I mentioned that I wished I knew who to credit for that remarkable routine to redefine the cursor. Dave Peden has written me that credit should be given to Terry L. Atkinson of 28 Savona Ct., Dartmouth, NS B2W 4R1 CANADA.

And I would like to strongly recommend that you support the 99'ers Users Group Association, 3535 So. H st., #93, Bakersfield CA 93304. They are a strictly non-profit group, devoting a lot of time and effort to helping us all, and they publish a great newsletter..

Every Tips must include a bit of music, and my grandson has requested that I pass this one on to all other two-year olds.

```
100 !ALPHABET SONG - by Jim
Peterson
110 DIM N(21)
120 CALL MAJORSCALE("C",N())
130 CALL SCREEN(5):: DISPLAY
 AT(24,1)ERASE ALL:"READY -
TYPE THE ALPHABET" :: CALL M
AGNIFY(2)
140 CALL KEY(3,K,ST):: IF (S
T<1)+(K<65)+(K>90)THEN 140 :
: CALL SPRITE(#1,K,16,96,120
):: IF K=87 THEN GOSUB 220 E
LSE GOSUB 200
150 IF (K=90)*(FLAG=0)THEN 1
60 ELSE 140
160 FLAG=1 :: M$="C115566D5C
443322D1" :: T=150
165 FOR J=1 TO 18 :: CALL SP
RITE(#J,64+J,INT(11*RND+6),9
6,128,J*5,J*5)
170 X=ASC(SEG$(M$,J,1)):: IF
 X>58 THEN T=150*(X-64):: GO
TO 190
180 X=X-48 :: CALL SOUND(T,N
(X),0)
190 NEXT J :: FLAG=0 :: CALL
 DELSPRITE(ALL):: GOTO 140
200 Y=VAL(SEG$("115566544332
22215543325332",K-64,1))
210 CALL SOUND(500,N(Y),0)::
 RETURN
220 CALL SOUND(500,N(5),0)::
 CALL SOUND(500,N(5),5):: CA
LL SOUND(500,N(4),0):: RETUR
N
230 SUB MAJORSCALE(K$,N())
240 F=VAL(SEG$("110123131147
165175196",POS("ABCDEFG",K$,
1)*3-2,3))
250 C$="10101101010110101101
0101101011010101"
260 FOR J=1 TO 36 :: IF SEG$
(C$,J,1)="0" THEN 280
270 X=X+1 :: N(X)=F*1.059463
094^(J-1)
280 NEXT J :: SUBEND
```

Lines 230-280 of that routine are an example of the kind of handy-dandy subprograms you will find on my Nuts & Bolts disk.

We haven't had a Tigercub Challenge for some time, so -

How can you store a hundred or more values of any size, positive or negative, integer or non-integer, even in exponential notation, without dimensioning an array or opening a file?

Now, how can you link your program to another by a RUN statement, thereby losing all data, and recover those values? Yes, I know you can save them on the screen and read them back, but can you find a better way?

Here's a little demo program of how motion can be created by the repetitive redefinition of characters. I call it ETERNITY.

```
100 CALL CLEAR :: CALL SCREE
N(2):: CALL COLOR(1,16,1)::
CALL CHAR(33,"",34,"",35,"",
36,"")
120 FOR R=1 TO 12 :: CALL HC
```

```
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
BASE->R DECIMAL 16 SYSTEM 0 0 GOTOXY
 ."     FORTH Screens to/from V80 files     " CR
 ."              by  JWVincent              " CR
 ."  These screens will read or write TI    "
 ."  variable 80 files to or from TI-FORTH  "
 ."  screens. If DISK_HI equals DISK_SIZE   "
 ."  one drive will be used. When using one "
 ."  drive, begin with FORTH loaded, you    "
 ."  will be prompted when to load each     "
 ."  disk. If multiple drives are used place"
 ."  FORTH in # 1 and the V80 files disk in "
 ."  # 2.  V80 files read/written must/will "
 ."  be named SCRNxxx where xxx is the      "
 ."  screen number. When reading a V80 file "
 ."  EOF will cause a disk error, after     "                    -->
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
 ."  which the FORTH disk should be loaded  "
 ."  and the FLUSH command executed.        "
 ."  The word format is:                    " CR
 ."       n1 n2 WRITE-V80 ( n1=start scrn ) "
 ."          n1 READ-V80  ( n2= end  scrn ) "
68 CLOAD STAT  84 CLOAD message
0 VARIABLE DF  0 VARIABLE PF  0 VARIABLE BUFR 78 ALLOT
PABS @ 10 + BUFR 6144 FILE V80
V80 SET-PAB VRBL 80 REC-LEN F-D" DSKn.SCRNxxx"
: DRIVE-NO DISK_HI @ DISK_SIZE @ = IF 0 ELSE 1 THEN DF ! ;
: V80-DSK DF @ IF ELSE CR ." LOAD V80 DISK" CR KEY DROP THEN ;
: FTH-DSK DF @ IF ELSE ." LOAD FORTH DISK " CR KEY DROP THEN ;
: FIX-NAME DRIVE-NO PABS @ DF @ 49 +  OVER 23 + VSBW
   OVER 100 /MOD 48 + ROT 29 + VSBW 10 /MOD 48 +
   PABS @ 30 + VSBW 48 + PABS @ 31 + VSBW V80-DSK ;          -->
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
: R-LEN DUP 63 + 64 0 DO I OVER OVER - C@ 32 = IF I 63 < IF
    DROP THEN ELSE LEAVE THEN LOOP SWAP DROP 64 SWAP - ;
: WRITE-16 16 0 DO DUP BUFR 64 CMOVE R-LEN WRT 64 + LOOP DROP ;
: WRT-V80  CASE  1 OF 1 ENDOF  2 OF SWAP 2 ENDOF
                 3 OF SWAP ROT 3 ENDOF   4 OF SWAP ROT >R ROT R>
                   SWAP 4 ENDOF  ENDCASE
           0 DO WRITE-16 LOOP ;
: SCR-RD OVER OVER 4 + > IF DUP 4 + OVER 1 ELSE 0 THEN PF !
    0 ROT RDT DO I BLOCK SWAP 1 + LOOP ;
: WRITE-V80 1+ SWAP DUP >R SCR-RD R> FIX-NAME DROP OPN
    BEGIN WRT-V80 PF @
    WHILE FTH-DSK 4 + SCR-RD V80-DSK
    REPEAT CLSE ;

                                                              -->
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
: READ-4 8210 OVER 4 + ROT
      DO DUP 1024 32 FILL  BLANKS^
           16 0 DO RD DUP 64 > IF DROP 64 THEN
                I IF ELSE OVER J 32768 OR SWAP 2 - ! THEN
                    OVER BUFR SWAP ROT CMOVE 64 + LOOP
         4 + LOOP DROP FTH-DSK FLUSH ;
: READ-V80 FIX-NAME OPN BEGIN DUP READ-4 4 + V80-DSK AGAIN ;
EMPTY-BUFFERS R->BASE   ( Can you use =0 -TRAILING FIRST UPDATE
```

TI-FORTH's direct sector I/O is a very fast, and efficient method for handling screens within the FORTH environment. However, when you want to trade a couple screens with your friend (along with some other files), or send them via modem, it can be a real problem. These screens solve that problem by formatting screens to TI's standard variable 80 format. They also support reading screens from V80 files.

The screens contain instructions for their use, so, I will use this column to comment on the words defined and other items. First the screens show the instructions so that you can read while they load. Next, they insure that the -FILES and memory resident messages are loaded. I had to use these messages because the disk based ones make the screen buffers flush when READ-V80 encounters EOF. That's disasterous if your using a single drive since your V80 disk is still loaded! Anyway, I found the system very apt to hang with these messages loaded and I don't recommend using them if you have a choice. I also had to clear the buffers myself since the FORTH word CLEAR dumps each time it executes (I wanted to move more than 1 scrn at a time) Due to space and time constraints, the comments will be brief. You must study this to understand how it works.

Hint: read FORTH programs from the bottom up.

```
(                  Common Words                              )
( set up the PAB with a dummy filename                       )
( check for multiple drives and set Disk Flag                )
( if single drive display message and wait for key press     )
( ditto                                                      )
( get PAB addr and use Disk Flag to set disk number          )
( divide screen number by 100 then 10 and use the            )
(     results to fill in the disk name                       )
(                  Write Words                               )
( count trailing blanks in record and adjust record          )
(      length. minimum record length is one byte             )
( write the 16 records from the current scrn to V80 file     )

( reverse the stack order of the screen buffers read         )
(     and put number of buffers back on top of the stack     )
(     then loop thru writing each screen that was read       )
( if there are more than 4 scrns left to read set PF         )
( read up to 4 and leave their buffer addrs and count        )
( fix input, read first screens, set file name and open      )
( write the screens you read to V80 and check Pass Flag      )
( if there's more to read, prompt disk swap and read it      )
( if not close the file...  your done.  )

                                                             -->
(                  Read Words                                )
( put first scrn bufr addr, scrn num +4, scrn num onstack)
( for each screen bufr up to 4 clear the buffer then         )
(     read 16 records from the V80 file checking the len     )
(     after the first read, flag the screen as updated       )
(     move the actual number of bytes read to the screen     )
(     loop back for the rest, prompt disk swap and flush     )
( set file name, open it read 4 scrns/pass til EOF crash     )
and +BUF to shorten this program? Try it )
```

FORTH SCREENS TO OR FROM DISPLAY VARIABLE 80 FILES

By Jim Vincent

```
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
  TI-FORTH's direct sector I/O is a very fast, and efficient
BASE->R DECIMAL 16 SYSTEM 0 0 GOTOXY
  method for handling screens within the FORTH environment.
."     FORTH Screens to/from V80 files     " CR
  However, when you want to trade a couple screens with your
."             by  JWVincent               " CR
  friend ( along with some other files ), or send them via
."  These screens will read or write TI     "
  modem, it can be a real problem. These screens solve that
."  variable 80 files to or from TI-FORTH   "
  problem by formatting screens to TI's standard variable 80
."  screens. If DISK_HI equals DISK_SIZE    "
  format. They also support reading screens from V80 files.
."  one drive will be used. When using one  "
."  drive, begin with FORTH loaded, you     "
  The screens contain instructions for their use, so, I will
."  will be prompted when to load each      "
  use this column to comment on the words defined and other
."  disk. If multiole drives are used place"
  items. First the screens show the instructions so that
."  FORTH in # 1 and the V80 files disk in  "
  you can read while they load. Next, they insure that the
."  # 2.  V80 files read/written must/will  "
  -FILES and memory resident messages are loaded. I had to
."  be named SCRNxxx where xxx is the       "
  use these messages because the disk based ones make the
."  screen number. When reading a V80 file  "
  screen buffers flush when READ-V80 encounters EOF. That's
."  EOF will cause a disk error, after      "                    -->
  disasterous if your using a single drive since your V80
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
  disk is still loaded! Anyway, I found the system very apt
."  which the FORTH disk should be loaded   "
  to hang with these messages loaded and I don't recommend
."  and the FLUSH command executed.         "
  using them if you have a choice. I also had to clear the
."  The word format is:                     " CR
  buffers myself since the FORTH word CLEAR dumps each time
."        n1 n2 WRITE-V80 ( n1=start scrn ) "
  it executes ( I wanted to move more than 1 scrn at a time)
."           n1 READ-V80  ( n2= end  scrn ) "
  Due to space and time constraints, the comments will be
68 CLOAD STAT  84 CLOAD message
  brief. You must study this to understand how it works.
0 VARIABLE DF  0 VARIABLE PF  0 VARIABLE BUFR 78 ALLOT
  Hint: read FORTH programs from the bottom up.
PABS ! 10 + BUFR 6144 FILE V80
  (                  Common Words                          )
V80 SET-PAB VRBL 80 REC-LEN F-O" DSKn.SCRNxxx"
  ( set up the PAB with a dummy filename                   )
: DRIVE-NO DISK_HI ! DISK_SIZE ! = IF 0 ELSE 1 THEN DF ! ;
  ( check for multiple drives and set Disk Flag            )
: V80-DSK DF ! IF ELSE CR ." LOAD V80 DISK" CR KEY DROP THEN ;
  ( if single drive display message and wait for key press )
: FTH-DSK DF ! IF ELSE ." LOAD FORTH DISK " CR KEY DROP THEN ;
  ( ditto                                                  )
: FIX-NAME DRIVE-NO PABS ! DF ! 49 +  OVER 23 + VSBW
  ( get PAB addr and use Disk Flag to set disk number      )
  OVER 100 /MOD 48 + ROT 29 + VSBW 10 /MOD 48 +
  ( divide screen number by 100 then 10 and use the        )
  PABS ! 30 + VSBW 48 + PABS ! 31 + VSBW V80-DSK ;          -->
  (     results to fill in the disk name                   )
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
  (                  Write Words                           )
: R-LEN DUP 63 + 64 0 DO I OVER OVER - C! 32 = IF I 63 < IF
  ( count trailing blanks in record and adjust record      )
  DROP THEN ELSE LEAVE THEN LOOP SWAP DROP 64 SWAP - ;
  (      length, minimum record length is one byte         )
: WRITE-16 16 0 DO DUP BUFR 64 CMOVE R-LEN WRT 64 + LOOP DROP ;
  ( write the 16 records from the current scrn to V80 file )
: WRT-V80  CASE  1 OF 1 ENDOF  2 OF SWAP 2 ENDOF
                 3 OF SWAP ROT 3 ENDOF   4 OF SWAP ROT >R ROT R>
  ( reverse the stack order of the screen buffers read     )
                 SWAP 4 ENDOF  ENDCASE
  (     and put number of buffers back on top of the stack )
        0 DO WRITE-16 LOOP ;
  (     then loop thru writing each screen that was read   )
: SCR-RD OVER OVER 4 + > IF DUP 4 + OVER 1 ELSE 0 THEN PF !
  ( if there are more than 4 scrns left to read set PF     )
  0 ROT ROT DO I BLOCK SWAP 1 + LOOP ;
  ( read up to 4 and leave their buffer addrs and count    )
: WRITE-V80 1+ SWAP DUP >R SCR-RD R> FIX-NAME OPEN DPN
  ( fix input, read first screens, set file name and open  )
  BEGIN WRT-V80 PF !
  ( write the screens you read to V80 and check Pass Flag  )
  WHILE FTH-DSK 4 + SCR-RD V80-DSK
  ( if there's more to read, prompt disk swap and read it  )
  REPEAT CLSE ;
  ( if not close the file... your done. )


                                                            -->
( FORTH screens to/from variable 80 files - JWVincent - 8/15/84)
  (                  Read Words                            )
: READ-4 8210 OVER 4 + ROT
  ( put first scrn bufr addr, scrn num +4, scrn num onstack)
     DO DUP 1024 32 FILL
  ( for each screen bufr up to 4 clear the buffer then     )
          16 0 DO RD DUP 64 > IF DROP 64 THEN
  (     read 16 records from the V80 file checking the len )
              I IF ELSE OVER J 32768 OR SWAP 2 - ! THEN
  (     after the first read, flag the screen as updated   )
                  OVER BUFR SWAP ROT CMOVE 64 + LOOP
  (     move the actual number of bytes read to the screen )
       4 + LOOP DROP FTH-DSK FLUSH ;
  (     loop back for the rest, prompt disk swap and flush )
: READ-V80 FIX-NAME DPN BEGIN DUP READ-4 4 + V80-DSK AGAIN ;
  ( set file name, open it read 4 scrns/pass til EOF crash )
EMPTY-BUFFERS R->BASE
```

KEY CODE / TOKENIZED BASIC CODE CHART
Adapted from an original compilation by Don Donlan

COMMAND	ASCII	Hex	Key
Marks EOL	0	>00	
AID	1	>01	FCTN 7
BREAK/INTRUPT	2	>02	FCTN 4
DELETE CHAR	3	>03	FCTN 1
INSERT	4	>04	FCTN 2
QUIT/RESET	5	>05	FCTN =
REDO/REPEAT	6	>06	FCTN 8
ERASE A LINE	7	>07	FCTN 3
Cursor Left	8	>08	FCTN S
Cursor Right	9	>09	FCTN D
Cursor Down	10	>0A	FCTN X
Cursor Up	11	>0B	FCTN E
PROCEED	12	>0C	FCTN 6
Carriage Rtrn	13	>0D	ENTER
BEGIN	14	>0E	FCTN 5
BACK	15	>0F	FCTN 9
DLEscape[1]	16	>10	CTRL P
DC1 (X-ON)[1]	17	>11	CTRL Q
DC2[1]	18	>12	CTRL R
DC3 (X-OFF)[1]	19	>13	CTRL S
DC4[1]	20	>14	CTRL T
NAKnowledge[1]	21	>15	CTRL U
SYNc Idle[1]	22	>16	CTRL V
ETBlock[1]	23	>17	CTRL W
CANcel[1]	24	>18	CTRL X
Endof Meduim[1]	25	>19	CTRL Y
SUBstitute[1]	26	>1A	CTRL Z
ESCape[1]	27	>1B	CTRL .
File Separtr[1]	28	>1C	CTRL ;
Grp Separatr[1]	29	>1D	CTRL =
Cursor Char	30	>1E	
Edge Char	31	>1F	
Blank/Space	32	>20	Space
	33	>21	!
	34	>22	"
	35	>23	#
	36	>24	$
	37	>25	%
	38	>26	&
	39	>27	'
	40	>28	(
	41	>29	)
	42	>2A	*
	43	>2B	+
	44	>2C	,
	45	>2D	-
	46	>2E	.
	47	>2F	/
	48	>30	0
	49	>31	1
	50	>32	2
	51	>33	3
	52	>34	4
	53	>35	5
	54	>36	6
	55	>37	7
	56	>38	8
	57	>39	9
	58	>3A	:
	59	>3B	;
	60	>3C	<
	61	>3D	=
	62	>3E	>
	63	>3F	?
	64	>40	@
	65	>41	A
	66	>42	B
	67	>43	C
	68	>44	D
	69	>45	E
	70	>46	F
	71	>47	G
	72	>48	H
	73	>49	I
	74	>4A	J
	75	>4B	K
	76	>4C	L
	77	>4D	M
	78	>4E	N
	79	>4F	O
	80	>50	P
	81	>51	Q
	82	>52	R
	83	>53	S
	84	>54	T
	85	>55	U
	86	>56	V
	87	>57	W
	88	>58	X
	89	>59	Y
	90	>5A	Z
	91	>5B	[
	92	>5C	\
	93	>5D	]
	94	>5E	^
Underline	95	>5F	
Grave Accent	96	>60	`
	97	>61	a
	98	>62	b
	99	>63	c
	100	>64	d
	101	>65	e
	102	>66	f
	103	>67	g
	104	>68	h
	105	>69	i
	106	>6A	j
	107	>6B	k
	108	>6C	l
	109	>6D	m
	110	>6E	n
	111	>6F	o
	112	>70	p
	113	>71	q
	114	>72	r
	115	>73	s
	116	>74	t
	117	>75	u
	118	>76	v
	119	>77	w
	120	>78	x
	121	>79	y
	122	>7A	z
	123	>7B	{
	124	>7C	|
	125	>7D	}
	126	>7E	~
Delete Char	127	>7F	
Null	128	>80	CTRL .
ELSE	129	>81	CTRL A
::	130	>82	CTRL B
! (rem)	131	>83	CTRL C
IF	132	>84	CTRL D
GO	133	>85	CTRL E
GOTO	134	>86	CTRL F
GOSUB	135	>87	CTRL G
RETURN	136	>88	CTRL H
DEF	137	>89	CTRL I
DIM	138	>8A	CTRL J
END	139	>8B	CTRL K
FOR	140	>8C	CTRL L
LET	141	>8D	CTRL M
BREAK	142	>8E	CTRL N
UNBREAK	143	>8F	CTRL O
TRACE	144	>90	CTRL P
UNTRACE	145	>91	CTRL Q
INPUT	146	>92	CTRL R
DATA	147	>93	CTRL S
RESTORE	148	>94	CTRL T
RANDOMIZE	149	>95	CTRL U
NEXT	150	>96	CTRL V
READ	151	>97	CTRL W
STOP	152	>98	CTRL X
DELETE	153	>99	CTRL Y
REM	154	>9A	CTRL Z
ON	155	>9B	CTRL .
PRINT	156	>9C	CTRL ;
CALL	157	>9D	CTRL =
OPTION	158	>9E	CTRL 8
OPEN	159	>9F	CTRL 9
CLOSE	160	>A0	
SUB	161	>A1	
DISPLAY	162	>A2	
IMAGE[2]	163	>A3	
ACCEPT[2]	164	>A4	
ERROR[2]	165	>A5	
WARNING[2]	166	>A6	
SUBEXIT[2]	167	>A7	
SUBEND[2]	168	>A8	
RUN[2]	169	>A9	
LINPUT[2]	170	>AA	
	171	>AB	
	172	>AC	
	173	>AD	
	174	>AE	
	175	>AF	
THEN	176	>B0	CTRL 0
TO	177	>B1	CTRL 1
STEP	178	>B2	CTRL 2
Comma (,)	179	>B3	CTRL 3
Semicoln(;)	180	>B4	CTRL 4
Colon (:)	181	>B5	CTRL 5
Rt Paren)	182	>B6	CTRL 6
Lt Paren (	183	>B7	CTRL 7
Ampersand &	184	>B8	FCTN ,
	185	>B9	
OR[2]	186	>BA	FCTN /
AND[2]	187	>BB	CTRL /
XOR[2]	188	>BC	FCTN 0
NOT[2]	189	>BD	FCTN ;
=	190	>BE	FCTN 8
< Less Than	191	>BF	FCTN H
> Grtr Than	192	>C0	FCTN J
+ Plus	193	>C1	FCTN K
- Minus	194	>C2	FCTN L
* Times	195	>C3	FCTN M
/ Divide	196	>C4	FCTN N
^ Exponent	197	>C5	FCTN O
(undefined)	198	>C6	FCTN Y
Flag Quoted$	199	>C7	
Fl Unquoted$	200	>C8	
Flag Line No	201	>C9	
EOF	202	>CA	
ABS	203	>CB	
ATN	204	>CC	
COS	205	>CD	
EXP	206	>CE	
INT	207	>CF	
LOG	208	>D0	
SGN	209	>D1	
SIN	210	>D2	
SQR	211	>D3	
TAN	212	>D4	
LEN	213	>D5	
CHR$	214	>D6	
RND	215	>D7	
SEG$	216	>D8	
POS	217	>D9	
VAL	218	>DA	
STR$	219	>DB	
ASC	220	>DC	
PI[2]	221	>DD	
REC	222	>DE	
MAX[2]	223	>DF	
MIN[2]	224	>E0	
RPT$[2]	225	>E1	
	226	>E2	
	227	>E3	
	228	>E4	
	229	>E5	
	230	>E6	
	231	>E7	
NUMERIC[2]	232	>E8	
DIGIT[2]	233	>E9	
UALPHA[2]	234	>EA	
SIZE[2]	235	>EB	
ALL[2]	236	>EC	
USING[2]	237	>ED	
BEEP[2]	238	>EE	
ERASE[2]	239	>EF	
AT[2]	240	>F0	
BASE	241	>F1	
TEMPORARY	242	>F2	
VARIABLE	243	>F3	
RELATIVE	244	>F4	
INTERNAL	245	>F5	
SEQUENTIAL	246	>F6	
OUTPUT	247	>F7	
UPDATE	248	>F8	
APPEND	249	>F9	
FIXED	250	>FA	
PERMANENT	251	>FB	
TAB	252	>FC	
# (File Num)	253	>FD	
VALIDATE[2]	254	>FE	
2 MARK EOF	255	>FF	

- - - - - - - - -

Footnotes.

[1] Used in PASCAL. See Users Reference Guide p.III-2.

[2] Used in EXTENDED BASIC.