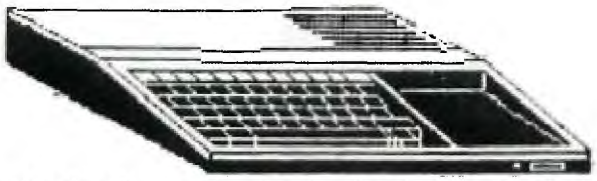
BYTE-LINE



(010)

MARCH 1989 8903

NEWSLETTER OF THE DECATUR 99ER USERS GROUP

STEVE THORPE...PRESIDENT
JESSE JOLLY...VICE PRESIDENT
KING FORKNER...TREASURER

SCOTTIE WILLIFORD...SECRETARY
CHARLES STRINGER...LIBRARIAN
GEORGE KORNFELD...EDITOR

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

PRESIDENT'S NOTES

THIS MONTH'S MEETING WILL BE ON MARCH 16, 1989 AT THE FIRST CONGREGATIONAL CHURCH. THE MEETING WILL BEGIN AT 6:30PM. AMPLE PARKING IS AVAILABLE ON THE SOUTH AND WEST SIDES OF THE CHURCH. AFTER THE PROGRAM IS PRESENTED, MEMBERS CAN COPY DISKS FROM THE LIBRARY AND SPEND TIME SOCIALIZING WITH OTHER MEMBERS. NEXT MONTH'S MEETING WILL BE ON APRIL 20, 1989.

THIS MONTH THE PROGRAM WILL BE ON THE PRINTER'S APPRENTICE TOOLBOX. THE PROGRAM WILL BE USING BORDERS SECTION OF MCCANN SOFTWARE, GIVEN BY GEORGE KORNFELD. HE WILL DEMONSTRATE A PRACTICAL APPLICATION FOR THIS SOFTWARE.
THE NEW VERSION OF TI-BASE VER.2.0 IS AVAILABLE FOR $7.95 +SHIP AND HANDLING. ALL MEMBERS THAT HAVE PURCHASED A TI-BASE WILL WANT THIS NEW UPDATE.

3.5" DRIVES ARE NOW AVAILABLE FOR THE TI-99/4A. THEY ARE FROM ALPHA SCIENTIFIC, PO BOX 626, CHESTERFIELD, MO 63006. PHONE, (314) 878-7117. THE COST: $93+SH.

FOR SALE: CONTACT STEVE THORPE - 217-422-9859
EXPANSION BOX, TI CONTROLLER, 2DSDD 1/2 HEIGHT DRIVES- $200
TI MULTIPLAN - $15 TI LOGO - $15 TI WRITER - $10 CARTRIGE EXTENDER - $15 PASCAL CARD AND MANUAL - $65
CARTRIGES- ADVENTURE, TI INVADERS, MUSIC MAKER, HOUSEHOLD BUDGET MANAGEMENT, HOME FINANCIAL DECISIONS, PERSONAL RECORD KEEPING, TEII- $20
1/2 HEIGHT DSDD DISK DRIVE - $45

EXPANSION BOX, TI CONTROLLER, 1SSSD DRIVE, RS232 CARD, 32K MEMORY CARD, CONSOLE $225

IF YOU HAVE NOT DONE SO, PLEASE RENEW YOUR MEMBERSHIP AT THIS MEETING OR SEND YOUR DUES TO OUR TREASURER, KING FORKNER. THE CLUB NEEDS YOUR SUPPORT TO CONTINUE. THANK YOU FOR THE GREAT SUPPORT AND PLEASE KEEP COMMING TO OUR MEETINGS.

STEVE THORPE

Minutes for Feburary 16,1989 Decatur 99ers

==========

****

Charles Stringer
Robert Walker
Scottie Williford
Ray Henderson
King Forkner
Jenelle Dearen
Charles Rothwell
Chris Kornfield
Aubrey Johnson
Bill Dearen
Jerry Rothwell
George Kornfield
James Freeman

OLD BUSSINESS:

Central Illinois Computerfest, gave check to 99ers, for, $50.00 as our part of the profits.

NEW BUSSINESS:

Liburary needs more input as to what is needed in our liburary. If you have not paid our dues please do so. We have $237.00, in the club treasury.

COMMING SOON:

March: TPA(PRINTER APPRENTICE) George Kornfield
April: DATA BASE 1, Scottie Williford.
May: TELECOMMUNICATION, Scottie Williford.
June: RAM DISK, Steve Thorpe.
July: CSDG, King Forkner.

PROGRAM:TI BASE, Charles Stringer, demonstrated, how a file was created, as usual Charles did a great job.

KEEP THEM COMMING(SEC.)

Meeting was adjourned at 2130hrs.
Respectively submited,
Scottie Williford
Feburary 16,1989

NEWS OF THE LIBRARY

ANOTHER VERSION OF 'MONOPOLY'--the Australian version by Ross Mudie, called 'TI-99-OPOLY'--was donated to the library at the last meeting. It's available for copying.

Also received--FUNNELWEB v4.13; this isn't likely to be of interest to the most of us, as the main change is a 'fix' of an incompatibility problem which occurs only if you're using BOTH a Horizon ramdisk and a Myarc disk controller. There has been released, by the same authors, an optional 80-column editor for use with the DIJIT AVPC card. There is a review of this editor in the Lima OH UG newsletter for February, as well as text-file prepared by DIJIT SYSTEMS concerning the hardware needed to generate and display an 80-column screen.

We received also from LIMA UG a neat utility for tape based systems, called CS1*FINDEX. This program, written in BASIC (not XBASIC) will search a specially-prepared program tape from the beginning until a specific program is found. (The program can't be used to read already-existing tapes; but programs could, of course, be read and re-recorded.) We hope that the Editor will be able to publish Charles Good's review in this issue of the newsletter. Although the program is being stored on disk, we will be glad to transfer it to tape for those who can't use disk. Please call me at 877-2780 if you want the tape version, and I'll have it ready for the meeting. There will be a charge of $1.25 for the cassette.

The LIMA disk also contains a MAIL LIST program written by Harry Allston, who is a correspondent of King Forkner's. Have you used this, King?

Because the existing disk is double-sided with some of the files in ARCHIVE'd format, I'm preparing a second version, which contains only the MAIL LIST and CS1*FINDEX programs and documents in SSSD format. Ask for LIMA/89/02.

We've also received a disk from the TIGERCUB with a final version of the Public Domain catalog, an index of the TIGERCUB's TIPS (361 entries!!), and a number of other things which I haven't time to describe now. Again, the distribution disk is DSSD, so we'll put a part of it (PD catalog and the TIPS INDEX) into the library this month as disk TICUB89/03.

If George can spare the space, we'll follow this column with a list of the TI-PD disk numbers and titles. The disk TICUB89/03 contains the title (and authorship, usually) of each program on each disk.

8 March 1989 C S Stringer

==========

TI-PD public domain software for the TI-99/4A computer, $1.50 per disk postpaid (minimum 8 disks please); number of sectors filled is indicated in parentheses). For a 9-page catalog listing all titles and authors, send $1 which is deductable from first order. (specify TI-PD catalog) Offered as a copying service only, without warranty other than that copies are equal to the original. Make checks payable to Tigercub Software (no credit card orders). Send to Tigercub Software, 156 Collingwood Ave., Columbus OH 43213.

600. Sam Moore Jr. Music #1 (341)
601. Sam Moore Jr. Music #2 (343)
602. Sam Moore Jr. Music #3 (348)
603. Sam Moore Jr. Music #4 (337)
604. Bill Knecht Hymns (334)
605. Christmas Music (318)
606. Holiday Music (339)
607. Great Songs by Bill Knecht (351)
608. Music by Bill Knecht (295)
609. March Music (329)
610. Tigercub Country Music (356)
611. Christmas Sing-Along (351)
612. J. Stephen Foster Music #1 (332)
613. J. Stephen Foster Music #2 (317)
614. Bach Music Programs (338)
615. Sing-Along Music (351)
616. Some of the Best Music (343)
617. Classical Music (340)
618. Assorted Music (346)
619. Chopin's Polonaise (280)
620. Assorted Music #2 (351)
621. Hamilton UG Music Package #1 (346)
622. Assorted Music #2 (349)
623. A Diskfull of J.S. Bach (345)
624. Assorted Music #4 (358)
625. Assorted Music #5 (347)
626. J.S. Bach Music (340)
627. Assorted Music #6 (354)
628. Some of the Very Best (349)
629. Ollie Hebert's Music (340)
630. Gregory Rashall Music Master (287)
631. Assorted Music #7 (352)
632. Sonata for Pianoforte (222)
633. Sonata for Pianoforte DS/SD
634. Strange Music (337)
635. Chuck Berry Tunes (218)
636. Christmas Songs w/Graphics (310)
637. Assorted Music #9 (337)
638. Classical Music #2 (338)
639. Assorted Music #10 (347)
640. Marches and College Songs (336)
641. Another Sing-Along (345)
642. Sing-Along Music #2 (236)
643. Christmas Music #2 (351)
644. Christmas Sing-Along #2 (340)
645. Classical Music #3 (352)
646. Assorted Music #11 (350)

647. Music Doodlers and Tinytunes (302)
648. Rhapsodie in Blue (287)
649. Assorted Music #8 (354)
650. Christmas Music w/Graphics 2 (357)
651. Sorgan II (145)
652. Christmas Music w/Graphics 3 (352)
653. Pop Demo V1.1 (225)
654. Christmas Music w/Graphics 4 (255)
655. Assorted Music #12 (349)
701. Musical Education (350)
702. Musical Education #2 (318)
703. Musical Education #3 (188)
710. American Flags (360)
711. Flags of the World (345)
712. Geography - U.S. States (341)
713. Geography - U.S. States #2 (212)
714. World Geography (179)
730. American History (48)
750. Alphabet w/Speech (343)
751. Children's Programs w/speech (357)
752. Alphabet for Preschool (329)
753. Children's Prog. w/Speech #2 (335)
755. Shapes, Colors, Directions (173)
760. Spelling (324)
770. Vocabulary and Reading (293)
780. Preschool Math (341)
790. Elementary Addition, Subtract(257)
791. Addition & Subtraction (337)
796. Multiplication, Division (348)
797. Multiplication, etc. (224)
800. Higher Math (355)
801. Higher Math #2 (228)
810. Typing Practice (223)
815. Morse Code Teacher (155)
820. Health (354)
821. Health #2 (145)
830. Physics (111)
840. Nature (277)
850. Chemistry (277)
860. Astronomy (342)
861. Astronomy #2 (304)
870. Religion (346)
871. Religion #2 (42)
890. Teacher's Helpers (203)
900. Home Utilities (351)
901. Home Utilities #2 (342)
902. Home Utilities #3 (350)
907. Screen Drawing, Doodling (160)
909. High-Resolution Drawing (287)
910. Charts & Graphs (178)
912. Calculators & Converters (345)
913. Calculators & Convert. #2(147)
915. Financial Math (339)
916. Financial Programs (356)
918. Checkbook Programs (203)
920. Business Programs (146)
950. Genealogy
970. Astrology, Numerology etc. (171)
980. Radio Utilities (220)
990. Sports Programs (329)
1100. Character & Sprite Editors(254)
1101. Programmer's Utilities (346)
1102. Sorts, Scrambles, Searches (228)
1105. Auto-loaders (217)
1106. Disk Catalogers (268)
1107. Character Sets etc. (353)
1110. Assembly Utilities (357)
1111. Assembly Utilities, Routines(328)
1112. New Horizon Assembly Util. (269)
1119. Hardware Utilities (169)
1120. Sound Effects (197)
1130. Disk Labels & Jackets (284)
1131. Gemini Printer Utilities (224)
1132. Word Processing Utilities (182)
1133. Banners, Graphs, etc. (203)
1135. Speech Utilities & Demos (355)
1140. Music Composers (288)
1141. Assembly Music Compiler (265)
1145. Telecommunications Aids (342)
1150. Programming Tutorials (348)
1160. Assembly Tutorials #1 (231)
1161. Assembly Tutorials #2 (289)
1162. Assembly Tutorials #3 (357)
1163. Assembly Tutorials #4 (358)
1164. Assembly Tutorials #5 (340)
1300. Mathematical Games (133)
1301. Brain Games #1 (344)
1302. Brain Games #2 (345)
1303. Brain Games #3 (352)
1304. Brain Games #4 (352)
1305. Two-Player Brain Games (335)
1306. Brain Games #5 (345)
1307. Master Mind (322)
1310. Memory Games (235)
1315. Sargon Chess (155)
1320. Mazes #1 (342)
1321. Maze Games #2 (346)
1322. Maze Games #3 (338)
1330. Hangman Games (335)
1331. Wheel of Fortune #1 (249)
1332. Wheel of Fortune #2 (248)
1333. Word Games (310)
1340. Games by Roland Trueman (333)
1350. Card Games #1 (352)
1351. Card Games #2 (348)
1352. Card Games #3 (94)
1356. Dice Games (354)
1360. Board Games (321)
1361. Bingo (73)
1362. Checkers (238)
1363. Board Games #2 (287)
1367. Gambling Games (237)
1381. Bowling (289)
1382. Golf (138)
1383. Billiards, Boxing, etc. (250)
1400. Adventure Disk #1 (360)
1401. Adventure Disk #2 (306)
1402. Adventure Disk #3 (329)
1403. Adventure Disk #4 (324)
1415. Hammurabi Games (268)
1416. Text Games #1 (313)
1417. Text Adventures (340)
1425. Graphics/Text Adventures (354)
1426. Graphics/Text Adv. #2 (322)
1427. Graphics/Text Adv. #3 (325)
1430. Road Race Games (356)
1431. Keyboard Maneuvering (349)
1432. Road Crossing Games (344)
1433. Road Crossing Games #2 (175)
1434. Keyboard Games (347)
1435. Keyboard Maneuvering #2 (354)
1436. Slot Machines (343)
1437. Keyboard Games #2 (353)
1438. Keyboard Games #3 (343)
1440. Q*Bert Games (288)
1445. King Kong Type Games (351)
1455. Assembly Games (231)
1456. Assembly Games #2 (346)
1460. Children's Programs (345)
1461. Fun Games for Kids (353)
1462. Easy Games for Kids (346)
1470. Great Games (342)
1471. Assorted Games #1 (348)
1472. Assorted Games #2 (343)
1473. Texas Games Medley w/speech(346)
1474. Sea Battle Games (329)
1475. Joystick Games (342)
1476. Joystick Games #2 (355)
1477. Joystick Games #3 (346)
1478. Joystick Games #4 (338)
1479. Two-Player Joystick Games (353)
1480. Two-Player Keyboard Games (353)
1481. Joystick Games #5 (345)
1500. Kaleidoscopes & Displays (262)
1501. Sprite Displays (200)
1505. Poetry, Prose & Nonsense (128)

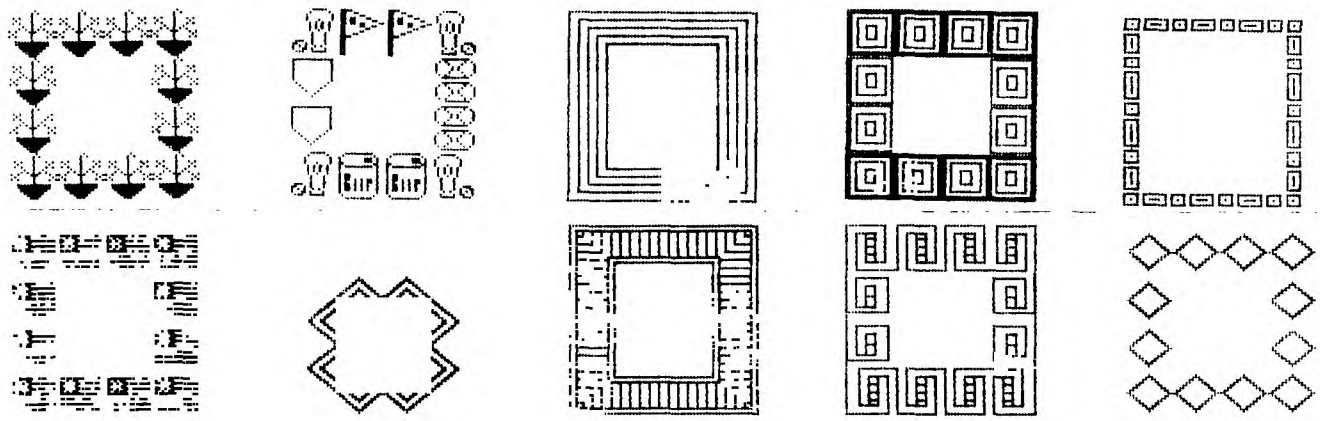
THE USE OF BORDERS FROM "TPA"

With the use of TPA TOOLBOX, of MCCANN Software, one can add a little something to a page, picture, or title. The something is a border. The border can be customized to enclose a small part to an entire page. There are two border fonts in TPA TOOLBOX. Each contain ten different borders and each one can be changed by use variables controls.

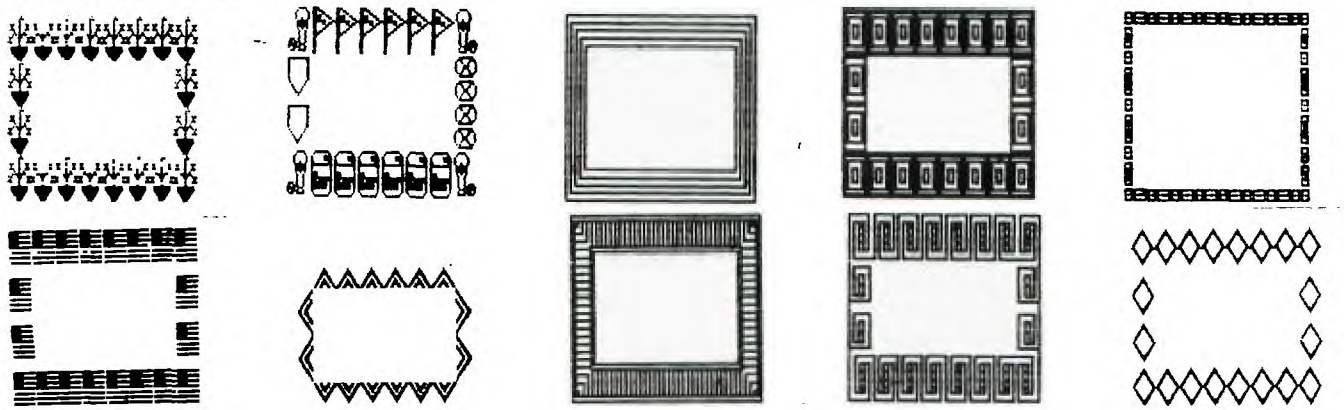
The variable controls are density of the print, font style, extra dark,line feed size, intercharacter width, vertical size and horizontal size. Some of the controls are line feed, the vertical distances between carriage return (1/216 th for Epson and 1/144 for Gemini). The intercharacter width is the distance between the horizontal elements of the border. Pixel width and height are computed automatically. The horizontal and vertical size are used to set the size of the border needed. The size of the border can be varied to a quarter of an inch.

In order to use the border font, first what size of font is needcd in inches. Then setup the variables needed for project. Then select the border font that fits the needs of project. Then dump the border font to a printer to check size and style picked. When satified with border printed, then dump to file to disk to be used with PRINTER'S APPRENTICE software. Then the border can be called up with the Scheduler Program.

SINGLE DENSITY



DOUBLE DENSITY



Reprinted from LIMA OH Newsletter BITS, BYTES & PIXELS/V.5, #2, pp 1-2/Feb 1989

CS1*FINDEX: AN AUTOMATIC CASSETTE TAPE PROGRAM LOCATION SYSTEM

A review by Charles Good

This one is for cassette tape users and for those interested in unusual programming techniques. Have you ever wondered if it was possible to mark with software the position of a specific program on a cassette tape full of many programs and then have the computer search the tape from the beginning until the specific desired program is found? TI did once develop such a system for its 99/8 computer, but TI's WAFERTAPE drive was never released. Coleco ADAM computers sccessfully use such a system. Not so for the TI99/4A, according to many well respected comentators. I have read again and again in our exchange newsletters expert comment to the effect that with the TI there is no way to automatically, under software control, advance a long cassette tape to the exact physical location where a program starts. Well....., way back as early as 1983 Joseph E. Bartle of Parish NY wrote a TI BASIC program that does this for the TI! I recently acquired a copy 1985 update of Joe's CS1*FINDEX program (still entirely in TI BASIC with no assembly routines) and after removing a few bugs I am quite impressed with capability of this software.

CS1*FINDEX will do its stuff even if you don't have a printed list of which programs are on a program tape, even if you are using a tape recorder that does not have a numerical tape counter, and even if you are using a tape recorder that is not automatically controlled on/off by the 99/4A. CS1*FINDEX finds semiautomatically the exact location of a program on a long tape. The manual tape recorder operations required of the user are all prompted from the screen. If you are using a TI compatible recorder, CS1*FINDEX will advance the tape to your program's location after you press fast foreward, and then automatically stop the tape. If you are using a tape recorder that the TI cannot automatically turn on and off, CS1*FINDEX will turn the screen from green to yellow and finally to red to indicate when you should manually press cassette STOP once the location of your program has been reached. Neat!

With CS1*FINDEX you can create a catalog of up to 10 programs you want to put on one side of a C60 tape and put this catalog at the beginning of the tape. The catalog includes program name (up to 12 characters with spaces anywhere), and there is also provision for catalog to display a 12 character comment for each of the 10 programs. You can then put your up to 10 programs onto the tape, with CS1*FINDEX advancing the tape recorder to the correct tape location where you should SAVE CS1 each program. It is necessary to reload CS1*FINDEX for each of the programs you put on the tape. Thus, users with only a console/cassette system will appreciate the fact that CS1*FINDEX is designed to be small enough to load into the MINIMEMORY module with SAVE MINIMEM. Then each time you need to load CS1*FINDEX, all you do is type OLD MINIMEM, and CS1*FINDEX boots in a few seconds. Otherwise it takes about 90 seconds to load CS1*FINDEX from tape.

Later, when you want to use the tape you load CS1*FINDEX into the computer and then load the tape's catalog from CS1*FINDEX. From the catalog display you select the number of the desired program on the tape. You are then instructed to rewind the tape to the beginning and press FAST FORWARD. CS1*FINDEX then advances the tape to the program's location, automatically stops the tape if you are using a TI compatible recorder, displays the name of your program on the screen, and informs you this program has been located. Then CS1*FINDEX BREAKs to command mode and allows you to load your program in the normal way by typing OLD CS1 and following all the usual screen instructions, except that you DO NOT again "rewind cassette tape". CS1*FINDEX can easily be modified in extended basic to load the located tape program into the computer from within CS1*FINDEX rather than from command mode. Change line 1770 to read RUN "CS1".

If you already have a printed list of each program on the tape and in which order the programs occur, you can bypass the catalog loading procedure. When you RUN CS1*FINDEX your first option is "LOCATION SEARCH (Y/N)". From here you can use CS1*FINDEX to locate the first or second or third, etc., program on the tape without using time to boot the catalog.

What's the secret? How does CS1*FINDEX using only TI BASIC with no assembly routines do what all the experts say can't be done? Have you ever noticed how the tape recorder behaves when you read or write tape serial FILES (as opposed to PROGRAMS)? The recorder starts, reads in or writes what I presume to be a file header, then stops. Then the recorder starts again and reads or writes the first record and then stops. Then the recorder starts again and reads or

writes the second record and then stops, etc., etc. The total number of start/stop cycles equals the number of records plus one. The computer controls the turning on and off of the tape recorder motor and IT DOESN'T MATTER TO THE COMPUTER IF THE RECORDER IS SET FOR PLAY OR FOR FAST FORWARD. When searching for a program, CS1*FINDEX writes a false file to the tape, turning the tape recorder motor on and off several times as this file is written. The tape recorder is set for FAST FORWARD rather than for RECORD as this file is written, so the tape never receives any data. The computer cannot directly sense that the tape is not getting any data, so the computer continues to turn the recorder motor on and off as it writes its fake file to the tape. When turned on, the tape advances very rapidly because the recorder is set for FAST FORWARD. A tape file designed to write up to 10 records with a record length of 192 will go through up to 11 start/stop sequences on a C60 tape before the tape is completely wound up on the take up reel. This is how CS1*FINDX locates physical blocks of tape space in which to insert programs, and can later find a specific program located at any one of these physical blocks of tape space. The first block (corresponding to the false file's header) is where the catalog is stored, and the next 10 blocks (each corresponding to a false file record) are where the programs are stored. Enough space is included in each of the program storage blocks to store the largest possible tape PROGRAM.

LIMITATIONS:

1--You can't use CS1*FINDEX with already existing program filled tapes. The spacing of the programs on the tape won't be right. You need to load programs onto your program storage cassette tapes using CS1*FINDEX.

2--Problems may occur if different tape recorders are used to store and later play programs. If the FAST FORWARD speed of the two recorders differs very much CS1*FINDX will not correctly find the location of the desired program.

3--There is only room for a short program in the last (10th) program block before the tape runs out.

The author of CS1*FINDEX has written some rather wordy documentation files to explain the use of CS1*FINDEX. These files are in PROGRAM format so that they can be loaded from tape and read by console/cassette-only users. In general most users can play around with the program and figure out how to use it without these docs. A sample tape program finding catalog is printed below as is the CS1*FINDEX program listing (checksums added using EZ-KEYS PLUS) with permission of the author Joseph E. Bartle. It is released to the TI community as FAIRWARE. If you like it, send whatever you think it is worth to Joe at the address in the REM statements at the beginning of the program. Joe has other fairware offerings. Write or call him for details. User groups, not individuals, may obtain a copy of CS1*FINDX and the above mentioned doc files by sending a disk and paid return mailer to the Lima User Group, P.O. Box 647, Venedocia OH 45894

SAMPLE FINDEX CASSETTE CATALOG

```
NUM     PROGRAMS

 1 ~ 3D TICTACTOE
 2 ~ BASEBALLSTAT
 3 ~ DRAW--------
 4 ~ FUN HOUSE___
 5 ~ MEMORY JOB--
 6 ~ SPELL QUIZ__
 7 ~ GOLFHANDICAP
 8 ~ LIGHT YEARS-
 9 ~ PHOTO DIARY-
10 ~ ------------
REMARKS!
11 R TIB---------
12 R TIB/DATAFILE
13 R TIB TEACHING
14 R XB/JS OPTION
15 R TIB/DATAFILE
16 R TIN/DATAFILE
17 R TIB---------
18 R TIB---------
19 R TIB---------
20 R ------------
```

TIPS FROM THE TIGERCUB

#55

Tigercub Software
156 Collingwood Ave.
Columbus OH 43213

The Tigercub has dipped a cautious paw into the cold dark mysterious waters of assembly, while still keeping a firm grip on trusty old Extended Basic. The result is an XBasic program that writes an assembly program!

The following subprogram, when merged into any program which has reidentified characters, and called after the characters have been reidentified, will write a source code which can be assembled into object code, loaded from XBasic and linked to instantly access the character set.

The source code is based on 2FONTS/S by Barry Traver, who gives credit to Mac McCormick, David Migicovsky and Karl Schuneman.

```
19000 SUB CHARSUB(HX$())
19001 DISPLAY AT(12,1)ERASE
ALL:"Source code filename?":
"DSK" :: ACCEPT AT(13,4)SIZE
(12)BEEP:F$ :: OPEN #1:"DSK"
&F$,OUTPUT
19002 DISPLAY AT(15,1):"LINK
ABLE program name?" :: ACCEP
T AT(16,1)SIZE(6):P$
19003 DISPLAY AT(18,1):"Rede
fine characters from ASCI
I to ASCII"
19004 ACCEPT AT(19,7)VALIDAT
E(DIGIT)SIZE(3):F
19005 ACCEPT AT(19,21)VALIDA
TE(DIGIT)SIZE(3):T
19006 PRINT #1:TAB(8);"DEF";
TAB(13);P$ :: PRINT #1:"VMBW
  EQU  >2024" :: PRINT #1:"
STATUS EQU  >837C"
19007 NB=(T-F+1)*8 :: CALL D
EC_HEX(NB,H$):: A=768+F*8 ::
 CALL DEC_HEX(A,A$)
19008 FOR CH=F TO T :: IF CH
<144 THEN CALL CHARPAT(CH,CH
$)ELSE CH$=HX$(CH)
19009 IF FLAG=0 THEN PRINT #
1:"FONT";:: FLAG=1
19010 FOR J=1 TO 13 STEP 4 :
: M$=M$&">"&SEG$(CH$,J,4)&",
" :: NEXT J :: M$=SEG$(M$,1,
23)&"  *"&CHR$(CH)
19011 PRINT #1:TAB(8);"DATA
"&M$ :: M$="" :: NEXT CH
19012 PRINT #1:P$;TAB(8);"LI
  R1,FONT" :: PRINT #1:TAB(
8);"LI  R0,>"&A$ :: PRINT #
1:TAB(8);"LI  R2,>"&H$
19013 PRINT #1:TAB(8);"BLWP
@VMBW":TAB(8);"CLR @STATUS"
:TAB(8);"RT":TAB(8);"END" ::
 CLOSE #1
19014 SUBEND
19015 SUB DEC_HEX(D,H$)
19016 X$="0123456789ABCDEF"
:: A=D+65536*(D>32767)
19017 H$=SEG$(X$,(INT(A/4096
)AND 15)+1,1)&SEG$(X$,(INT(A
/256)AND 15)+1,1)&SEG$(X$,(I
NT(A/16)AND 15)+1,1)&SEG$(X$
,(A AND 15)+1,1):: SUBEND
```

Now to try it out. You probably know that CALL CHARSET will restore reidentified characters below ASCII 96 to normal form, but not those above, so let's write a routine to restore those. Clear the memory with NEW, merge in the above, which you should have SAVED with -

SAVE DSK1.CHARSUB,MERGE

by MERGE DSK1.CHARSUB. Add a line -

100 CALL CHARSUB(HX$()) and RUN. Answer the filename prompt with DSK1.OLDLOW/S, the next prompt with OLDLOW and select ASCII 97 to 127.

When done, insert the Editor/Assembler module and its disk Part A. Select Assembler, Y to load assembler, give the source code DSK1.OLDLOW/S, object code DSK1.OLDLOW/O, just press Enter at next prompt, and R for options. You should get 0000 ERRORS.

Now key in this routine to test your program.

```
100 CALL INIT :: CALL LOAD("
DSK1.OLDLOW/O"):: FOR CH=33
TO 126 :: CALL CHAR(CH,"FF81
8181818181FF"):: PRINT CHR$(
CH);:: NEXT CH
101 CALL KEY(0,K,S):: IF S=0
 THEN 101 ELSE CALL CHARSET
102 CALL KEY(0,K,S):: IF S=0
 THEN 102 ELSE CALL LINK("OL
DLOW")
110 GOTO 110
```

Press any key to restore the upper case characters by CALL CHARSET, any key again to use the CALL LINK.

You are now ready to use the routine to copy all kinds of character sets from the programs in your library. You don't have any such programs? Not to worry. You don't have to reidentify characters one by one with one of those graphics editor programs. You can just manipulate the existing hex codes of the normal characters. I have created nearly 50 different character sets by that method!

The space occupied by a character on the screen is really an 8x8 square of 64 tiny dots. Various dots are turned on (colored) and off (transparent) to create a pattern - just the opposite of light bulbs on a scoreboard.

And those on-and-off dots are really the binary numbers which the computer uses. But fortunately the computer lets us use hexadecimal numbers rather than binary. The following will print out a reference chart of decimal to binary to hexadecimal. You can easily convert it to dump to a printer.

```
10 DISPLAY AT(6,1)ERASE ALL:
"DEC BIN HEX"
100 FOR J=0 TO 15 :: CALL DE
C_BIN(J,B$):: CALL DEC_HEX(J
,H$):: DISPLAY AT(J+8,1):J;T
AB(5);B$;TAB(10);SEG$(H$,4,1
):: NEXT J
21020 SUB DEC_BIN(D@,B$):: D
=D@ :: IF D=0 THEN B$="0000"
 :: SUBEXIT
21021 IF D=1 THEN 21022 :: X
=D/2 :: B@$=STR$(ABS(X<>INT(
X)))&B@$ :: D=INT(X):: IF D>
1 THEN 21021
21022 B@$="1"&B@$ :: B$=RPT$
("0",4-LEN(B@$))&B@$ :: B@$=
"" :: SUBEND
21039 SUB DEC_HEX(D,H$)
21040 X$="0123456789ABCDEF"
:: A=D+65536*(D>32767)
21041 H$=SEG$(X$,(INT(A/4096
)AND 15)+1,1)&SEG$(X$,(INT(A
/256)AND 15)+1,1)&SEG$(X$,(I
NT(A/16)AND 15)+1,1)&SEG$(X$
,(A AND 15)+1,1):: SUBEND
```

And this routine will show you how each letter is formed, by binary 0's (off) and 1's (on), for each key you press. I put it in merge format so you can MERGE it into any program and CALL it to examine the characters.

```
17000 SUB CHARVIEW
17001 !programmed by Jim Pet
erson Feb 1989
17002 DISPLAY AT(1,1)ERASE A
LL:"CHARACTERS IN BINARY & H
EX":::"Press any key to see
the binary representation
of the screen character and
```

```
its hexcode."
17003 DISPLAY AT(8,1):"Press
 Enter to see the char-acter
."
17004 CALL KEY(0,K,S):: IF K
=13 THEN 17005 ELSE IF S=0 O
R K<32 OR K>143 THEN 17004 E
LSE 17007
17005 CALL CHAR(48,"FF"&RPT$
("81",6)&RPT$("FF",9))
17006 CALL KEY(0,K,S):: IF S
<1 THEN 17006 ELSE CALL CHAR
(48,"003844444444438001030
1010101038")::  GOTO 17004
17007 CALL CHARPAT(K,CH$)
17008 R=12 :: FOR J=1 TO 15
STEP 2
17009 H$=SEG$(CH$,J,1):: CAL
L HEX_BIN(H$,B$)
17010 DISPLAY AT(R,8):B$
17011 H$=SEG$(CH$,J+1,1):: C
ALL HEX_BIN(H$,B$)
17012 DISPLAY AT(R,12):B$ ::
 DISPLAY AT(R,18):SEG$(CH$,J
,2):: R=R+1 :: NEXT J :: DIS
PLAY AT(22,6):CH$ :: GOTO 17
004
17013 SUBEND
17014 SUB HEX_BIN(H$,B$):: H
X$="0123456789ABCDEF" :: BN$
="0000X0001X0010X0011X0100X0
101X0110X0111X1000X1001X1010
X1011X1100X1101X1110X1111"
17015 FOR J=LEN(H$)TO 1 STEP
 -1 :: X$=SEG$(H$,J,1)
17016 X=POS(HX$,X$,1)-1 :: T
$=SEG$(BN$,X*5+1,4)&T$ :: NE
XT J :: B$=T$ :: T$="" :: SU
BEND
```

And to reidentify a character, you just change the numbers and letters in the 16-digit hex code which represents the binary pattern. By writing little routines to switch those digits around, all kinds of things can be done.

For instance, the normal characters always have the top row of dots turned off, to provide spacing between lines of text on the screen. If you want taller characters you will have to double-space the lines, but you can create them by making the numerals and upper case characters consist of the 2nd-7th rows, the 7th row again, and the 8th row - it just happens to work out.

```
18000 SUB HIGHCHAR :: FOR CH
=48 TO 90 :: CALL CHARPAT(CH
,CH$):: CALL CHAR(CH,SEG$(CH
$,3,10)&RPT$(SEG$(CH$,13,2),
2)&SEG$(CH$,15,2)):: NEXT CH
 :: SUBEND
```

I made that a subprogram so you can MERGE it in and use it to modify other character sets.

If we take the hex code apart, 2 digits at a time, and reassemble it backward,

```
100 CALL CLEAR :: FOR CH=33
TO 90 :: CALL CHARPAT(CH,CH$
):: FOR J=1 TO 15 STEP 2 ::
CH2$=SEG$(CH$,J,2)&CH2$ :: N
EXT J :: CALL CHAR(CH,CH2$):
: CH2$="" :: NEXT CH
110 DISPLAY AT(12,1):"?NWOD
EDISPU":"VT EHT DENRUT OHW !
YEH" :: GOTO 110
```

That one was in my first Tips newsletter, years ago, but it is much more effective at assembly speed.

This one shades characters on their left edge by turning on the pixel to the left of the leftmost "on" pixel, if any. Also try it in combination with HIGHCHAR.

```
18001 SUB NEWCHAR3 :: FOR CH
=48 TO 122 :: CALL CHARPAT(C
H,CH$):: FOR J=1 TO 15 STEP
2
18002 CH2$=CH2$&SEG$("0367CD
EF",POS("01234567",SEG$(CH$,
J,1),1),1)&SEG$(CH$,J+1,1)::
 NEXT J :: CALL CHAR(CH,CH2$
):: CH2$="" :: NEXT CH :: SU
BEND
```

This one uses HIGHCHAR to heighten the character and then blanks out three rows. Try following it with NEWCHAR3.

```
18030 SUB NEWCHAR10 :: A$="0
0" :: FOR CH=48 TO 90 :: CAL
L CHARPAT(CH,CH$):: CH$=SEG$
(CH$,3,10)&RPT$(SEG$(CH$,13,
2),2)&SEG$(CH$,15,2)
18031 CH$=SEG$(CH$,1,4)&A$&S
EG$(CH$,7,2)&A$&SEG$(CH$,11,
2)&A$&SEG$(CH$,15,2):: CALL
CHAR(CH,CH$):: NEXT CH :: SU
BEND
```

The next one, which works only on ASCII 97-122, makes tall characters ridiculously elongated above.

```
18050 SUB NEWCHAR20 :: FOR C
H=97 TO 122 :: CALL CHARPAT(
CH,CH$):: CALL CHAR(CH,SEG$(
CH$,7,2)&RPT$(SEG$(CH$,9,2),
4)&SEG$(CH$,11,6)):: NEXT CH
 :: SUBEND
```

This one has the chaacters raised by one line, widened one column at left and two columns at right to make a full 8x8 character which must be double-spaced horizontally and vertically.

```
18090 SUB NEWCHAR27 :: FOR C
H=48 TO 122 :: CALL CHARPAT(
CH,CH$):: CH$=SEG$(CH$,3,10)
&RPT$(SEG$(CH$,13,2),2)&SEG$
(CH$,15,2):: FOR J=1 TO 15 S
TEP 2
18091 CH2$=CH2$&SEG$("014589
CD",POS("01234567",SEG$(CH$,
J,1),1),1)&SEG$("0129",POS("
048C",SEG$(CH$,J+1,1),1),1)
18092 NEXT J :: CALL CHAR(CH
,CH2$):: CH2$="" :: NEXT CH
 :: SUBEND
```

Those who have my Nuts & Bolts disks will see how valuable this assembly can be to make instantly available the routines for double height and double width characters, etc., etc. And if you have Todd Kaplan's amazing ALSAVE routine from the Genial Traveler Vol. 1 No. 3, you can imbed them in your XBasic program for fast loading.

And you can merge CHARSUB into any character editor or sprite defining program and, with a bit of modification, use it to convert your creations into fast-loading assembly.

These assembly loads are compatible with my BXB, so you can also load character sets into sets 15 and 16, ASCII 144-159. However, the CHARPAT statement cannot access ASCII above 143, so in this case you must dimension an array in the program you are copying from, as DIM HX$(159), and place the hex codes in the array using the ASCII as the subscript number, such as CALL CHAR(CH+64,CH$) :: HX$(CH+64)=CH$, so that they will be passed to the subprogram. And don't CALL INIT after you have called BXB!

So, now you try creating your own screen fonts!

Memory full,

Jim Peterson

TIME DATED MATERIAL

DECATUR 99er H.C.U.G.
P.O. BOX 726
DECATUR, IL 62525

NEXT MEETING DATE:
THURSDAY, MARCH 16, 1989

ALL MEETING DATES:
6:30 PM TO ???? PM
FIRST CONGRAGATIONAL CHURCH
3465 NORTH MacARTHUR RD.

>>>>>MARCH MEETING DATE<<<<<

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	8	10	11
12	13	14	15>>	16<<	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

DECATUR 99er HOME COMPUTER USERS GROUP
APPLICATION FOR MEMBERSHIP

DATE __/ __/ 89

NAME _____

ADDRESS _____

CITY _____ZIP_____

PHONE _____

WORK PHONE _____

DUES: MEMBERSHIP $20

STUDENT $15