THE HUGgers
HOOSIER USERS GROUP

DECEMBER, 1984 THE HUGgers NEWSLETTER VOLUME 2, NUMBER 9

THE OFFICER'S CORNER

I hope you all had a nice Thanksgiving. As promised, I will announce the winner of the Library Contest this month. The winner is Andy Armstrong, and the runner up is Jack Witt. Andy and Jack, you can pick up your prizes at the December Meeting. Congrats!

Instead of workshops at the December meeting, we're going to have a Christmas Party (as you can see from the info to the right). I'll go into a little more detail here about it. The doors will open at 1:00 pm. I will try to call the meeting to order promptly at 2:00 pm (for a change), and after I present Andy & Jack with their prizes, the dinner will begin. This will be a pitch-in type dinner with the Users Group furnishing fried chicken, ham, turkey, tea, lemonade and ice. Suggested items to bring are: veggies, casseroles, breads, cheeses, snacks, potato chips and last but certainly not least, desserts! All those bringing veggies and/or casseroles are requested to bring their own serving spoons (please label what you bring). Please call the phone number (to the right) if you plan on attending the meeting. Also, if you have any questions, please leave a message (including your name and phone number) and someone will get back with you.

Vic Kelson has informed me that he has an assembly language screen dump utility that he would like to demonstrate at this meeting. For those of you who are interested, please see Vic after the dinner. Vic has also started a monthly assembly language column in the Newsletter. While we're on the subject of assembly language, anyone who has an assembly language routine they would like to donate to the library, please save them on the NEW-PROGRAMS disk as follows: save the program with the Call Link name and /ASM, then write some instructions for your routine and save it using the Call Link name and /IN (preferably in DISPLAY/80 format). (Example: save a dump utility as DUMP/ASM & its instructions as DUMP/IN.) Vic has made a point, (and well taken) that the library is lacking in Editor/Assembler programs.

Several members have expressed an interest in obtaining a copy of the TI-FORTH Manual (a copy of which was furnished to the Users Groups by TI). We had 45 manuals printed and those are all sold. I would like to get 20 more manuals printed, but I need a minimum of 10 paid in full orders from the membership before I can authorize another printing. The FORTH manual have over 100 (double sided) pages and has a nice spiral "GBC" binding with an index card stock cover. Deposits for the manuals are $20 each (yes, you can order more than one) and your deposit will be refunded in the event 10 orders are not taken. If you would like to order a manual, please see Pam or myself at the next meeting.

See you at the Dec. 9th meeting. J. Steven Sims, Pres.

THE NEXT MONTHLY MEETING WILL BE
DECEMBER 9, 1984
STARTING AT 2:00 PM AT CREATIVE LOGIC.

COME HUNGRY!

FOR THE

CHRISTMAS PARTY!

(See below for details!)

CHRISTMAS PARTY INVITE!

The next meeting is going to be a social occasion. We're having a dinner party at the regular meeting time. The club will provide turkey, ham and fried chicken, along with tea and lemonade to drink and we would like you to bring a side dish to help round it off! (See the Officer's corner for more details).

We need to know if you're coming.....
So we've put an answering machine on a line so you can let us know! Please tell us:

* Who you are,
* How many are you bringing?
* What you plan to bring to eat,
* A phone number where we can reach you!

The number to call is **925-5283** and you can call **any time** night or day.

No demos are planned, but some of us will be bringing systems to play with.

SOUTH REIGONAL NEWS

Due to the holidays, there will be no December meeting. See Regional News for more South-Side News!

HAPPY BIRTHDAY!

A Happy Birthday to the following December, 1983 members!!!! Ralph Cooper, John Schneider, Waldron Elementary School (Carolyn Weintraut), Gordon Edwards, Gordon Goins, James Hawkins, David Artman, Clifford Krick, Karen Taber, Kurt Schoch, Lew Bartley, Dennis Joyce, Bob Brattain & family, Jerry Yarnell, Georgia Wyatt, Anthony Carlino, Terry Sullivan, Rene' Torrella, Dia Daniel, Michael Jancoviz, Ken Burrell, Peggy Davis, and John & Vickie Gould.

WELCOME!

A big Holiday Welcome to these new HUGgers who joined in the last month: Michael Lentz, Don Zimmerman, Darrell McConnell, Robert Stahlhut, Steve Seiler, and Johnny Powell.

WELCOME BACK!

And also a Welcome Back to these renewing HUGgers: Ed Ferguson, Paul Hubbard, Kirk Mangold, George Forest, George Gordon, Bill Lucid, Bill Jones (of Indy), Barb Uhrig, and Steve & Pam Sims.

REGIONAL NEWS

SOUTH REGIONAL MEETING

The South Regional group will not meet during December. It's a busy time for most of us. The library will still be available. Please call to arrange a time if you are interested (881-5918).

We will resume meeting again in January. One of our members has offered to give a presentation on program structure. Another member has offered to teach Extended Basic. If you are interested, please call me at the listed number.

BEST OF THE NEWSLETTER!
EXTENDED BASIC QUIRKS

Have you been trying to make the computer speak phrases using Extended BASIC & the Speech Synthesizer but are disappointed with the results? It is not documented that you need the # symbol before & after phrases that are listed (see the List Of Speech Words located in Appendix L of the Extended BASIC Manual). Phrases such as "WHAT WAS THAT", "READY TO START" & "THAT IS RIGHT" must be entered as CALL SAY("#WHAT WAS THAT#"), CALL SAY("#READY TO START#") & CALL SAY ("#THAT IS RIGHT#") in order to hear them spoken correctly.

Editor's Note: This is the last article of the year-long Best of the Newsletter series. This Best of the Newsleter appeared in the July, 1983 issue of the HUGger Newsletter, & was written by Ed York, of the Cin-Day Users Group, P.O. Box 519, West Chester, Ohio 45069.

MICROS IN ACTION
by Bill Cagle

Perceptions:

Recently, while talking to a computer hobbyist, I remarked about what an excellent disk manager TI had installed in the 99/4A. He seemed interested so I pressed on and started to recount the many blessing's of not having to reserve disk space prior to writing files or saving programs. I also told him that when we read in a program, enlarge it and then re-save the program, TI's "Disk Manager" would fill the old hole, fractionate the remainder and continue storing the remainder in the next available space. This person owns an IBM and when he heard this, he said "What price do you pay for this in time?" I told him that it was so quick that you couldn't tell the difference. He flushed, turned and walked away from me and I knew that while I had won a battle, I had hurt him badly. The previously mentioned encounter points up a danger in our hobby. When we talk to other people in our hobby, we should be aware that other people love their "hobby toys" as much as we love our's.

Each of us perceives our computer in a different light. The very existence of this club stems from the desire to learn and use the full potential of our computer to a higher degree, than we could do otherwise. Most people develop a fondness for their "toys" that far outreachs the utility value to them. This might be in part due to the enormous amount of time and money we must invest in our systems to bring them to their full potential.

I have the perception of my computer as a complete set of tools in a magic package; push a button and you have a hammer, push it again and you have a drill, push it again and you have a saw, push it still once more and you have a sander. This perception is more real than you might think, as your computer can change its identity by merely pushing a button (loading different software). This means that your magic tool needs software to change its identity, so keep up your efforts to write software (as well as purchasing it). To illustrate this, I recently purchased some software called "Multiplan" and when I tried to use it on my expenses I was dismayed to see how slow it was. This shows that a general purpose software is never as good as some written to do a specific job.

For all of you who are interested in Forth, I have found that the TI Doom series is written in Forth. If you have this game, you might want to look at them with the Forth Editor and study the programing style and methods. If you can figure out what the author is doing, you can increase your own programing knowledge.

/4A DISK FORMAT

Editor's Note: The following article was copied from "The Paper Peripheral", Newsletter of the Central Texas 99/4A Users Group; P.O. Box 3026; Austin, Texas.

We are indebted to Wayne Talbot, who found this information in a file on CompuServe. The information was contributed to CompuServe by Earl Hall, and we are grateful to him for the information. Wayne was looking for this information in order to get a good look at the disk for Tunnels of Doom, in order to figure out what is on that disk, to further his effort to write a new TOD adventure. (If anyone out there has any information that would help Wayne in this endeavor, please contact him through the Central Texas Users Group. All TOD fans will appreciate it!)

The following is a complete and, to the best of my knowledge, accurate description of the disk directory format and file storage allocation used by the 99/4A computer.

Sector 0 contains the Volume Information Block

Address	Contents
0000-0009	Disk name--up to 10 characters
000A-000B	Total number of sectors on disk ()0168=360,)02D0=720,)05A0=1440)
000C	)09 (# of sectors/trk)
000D-000F	'DSK' ()44534B)
0010	)50=Disk backup protected,)20=not protected
0011	# of tracks per side ()28=40,)23=35)
0012-0013	# of sides/density ()0101=SS/SD,)0201=DS/DD,)0202=DS/DD)
0038-end	Sector allocation bit map. See note below.

Note on)0038-end: This is a sector-by-sector bit map of sector use; 1=sector used, 0=sector available. The first byte is for sectors 0 through 7, the second for sectors 8 through 15, and so on. Within each byte, the bits is for sectors 0 through 7, the second for sectors 8 through 15, and so on. Within each byte, the bits correspond to the sectors from right to left. For example, if byte)0038 contained)CF00 then the first byte equals 1100 1111. This means that sectors 0 through 3 are used, sectors 4 and 5 unused and sectors 6 and 7 are used. Information for the 2nd side of a DS/DD disk starts at byte)0065 and ends at byte)0091.

Sector 1 contains the Director Link

Each 16-bit word lists the sector number of the File Descriptor Record for an allocated file, in alphabetical order of the file names. The list is terminated by a word containing)0000; therefore, the maximum number of files per disk is 127 [(256/2)-1]. If the alphabetical order is corrupted (by a system crash during name change, for instance), the binary search method used to locate files will be effected and files may become unavailable.

Sectors 2 to 21 contain the File Descriptor Records

Address	Contents
0000-0009	File name--up to 10 characters
000C	Filetype:)00=DIS/FIX,)01=Program (memory-image),)02=INT/FIX,)80=DIS/VAR,)82=INT/VAR File deletion protection invoked by Disk Manager 2 will be shown by)08 added to the above.
000D	# of (MAXRECSIZE) record/sector.
000E-000F	# of sectors allocated to this file. (Disk Manager 2 will list one more than this number, thereby including this sector in the sector count.)
0010	For memory-image program files and variable-length data files, this contains the number of bytes used in the last disk sector. This is used to determine end-of-file.
0011	MAXRECSIZE of data file.
0012-0013	file record count, but with the second byte being the high-order byte of the value.
001C-end	Block Link. See note below.

Note on file storage: Files are placed on the disk in first-come/first-served manner. The first file written will start at sector 0022, and each subsequent file will be placed after it. If the first file is deleted, a newer file will be written in the space it occupied. If this space isn't big enough, the file will be 'fractured', and the remainder will be placed in the next available block of sectors. The block link map keeps track of this fracturing. Each block link is 3 bytes long. The value of the 2nd digit of the second byte followed by the 2 digits of the first byte is the address of the first sector of the extent. The value of the 3rd byte followed by the first digit of the 2nd byte is the number of additional sectors within this extent. Sectors 2 through 21 are reserved for File Descriptor Records and are allocated for file data only if no other available sectors exist. If more than 32 files are stored on a disk, additional File Descriptor Records will be allocated as needed, one sector at a time from the general available sector pool.

THE ASSEMBLY LANGUAGE FORUM - STARTING OUT

By Vic Kelson

Hello, and welcome to what I hope will become a monthly feature of the HUG newsletter. As a HUGger for the last 9 months, I've noticed with dismay the lack of coordinated interest in assembly language. I do believe that there's interest, though, 'cause I've heard "boy, I'd like to learn assembly language, but I just never have" from several people. Here comes your chance -- it's not that hard... really!!!

About the "forum" -- as I said earlier, I want this to be a monthly feature. Most importantly, this is ALL OF OUR column. Please let me know of any ideas, corrections, comments, complaints, or IDIDITs (my word for the euphoric feeling when a sticky piece of code finally works) that you might have, either at the meetings, on the bulletin board, or through the mail. I am learning this stuff too, and I hope to help moderate an ongoing discussion about assembly language in the HUG. I will also be holding get-togethers at the meetings, so that anybody who has never tried assembly can learn from those who have. We'll start real simple, and gradually get more advanced as time goes by.

Enough of that. The first featured assembly program in the forum is (ta-daaa!) a fairly fast screen dump routine for TI BASIC. If you have a dot-matrix printer, you might have wanted to make a copy of that neat picture you drew on your screen, but it's real hard on the TI. There are screen dump utilities (callable from extended BASIC) on the market, one of which I saw at the last meeting. I even wrote one in BASIC, and they have one thing in common: THEY ARE S---L---O---WWW..... The routine in this article is a LOT faster. It is not ideal. In fact, I have an even better one already, but the purpose of the article is to show you how to speed up a BASIC program using assembly language.

The problem with a BASIC screen dump is making the characters come out right on the printer when you've got nifty redefined characters on the screen. You see, due to the difference between the tasks performed by the video generator in your console and the printer, the characters are described differently. For, say the letter A:

```
                7 8 8 8 8 7 0 0  <-- Printer definition
                F 8 8 8 8 F 0 0

 the dots are   X . . . . X . .  82   \
 zeroes, the    X . . . . X . .  82    \
 X's are ones   X . . . . X . .  82     \ TI definition
                X X X X X X . .  FC     / see CALL CHAR in
                X . . . . X . .  82    /  the BASIC manual
                X . . . . X . .  82   /
                X . . . . X . .  82  /
```

The hex codes at the top are the codes that the printer wants to see in hi-res graphics mode to make the character. The codes at right are the codes which define the character in the TI. Problem: how do we make the TI codes into printer codes??

The answer turns out to be simple in assembly language, because of an operation we can't do in BASIC, Pascal, FORTRAN... or most other high level languages: the SHIFT operation. Here's how it works for a "left shift"; If you have the following byte (8 bits):

```
0 0 1 1 0 1 0 0  (34 hex or 52 decimal, if
                  you're keeping track)
```

A left shift of 1 bit position will move all the bits one space to the left, resulting in:

```
0 1 1 0 1 0 0 0  (note that the rightmost bit
                  is filled with a zero)
```

(This is 68 hex or 104 decimal, 2 times the original value. Will moving the bits 1 space to the right divide by 2?? Try it -- why does it work??)

But how does this help us to change rows to columns, you ask? Well, you might have wondered what happens to the leftmost bit.. It falls out of the byte into a CPU flag, called the "carry bit". We can test the carry bit in our assembly program, and put either a zero or one, according to the flag, into our result byte.

Here's the algorithm for getting the leftmost column from a set of 8 row definitions: (I know you're completely lost by now, hang on...)

```
set the result byte to 0.
for I=1 to 8
  shift the result byte 1 bit to the left.
  get row defining byte number I from the group of 8.
  shift the row defining byte 1 bit to the left.
  if a one fell out (carry bit set):
    add one to the result byte (put the "1" bit in there)
  put back the shifted version of the row defining byte.
next I.
save the result byte. (it now contains the leftmost column)
```

Easy, huh?? For an exercise, draw the character definition for the letter A shown earlier on paper, and do the above algorithm 8 times. REMEMBER to put the shifted defining bytes back, so that if you loop through 8 times, the 8 result bytes will be printer definitions shown above. It really IS easy, I promise.

The algorithm I've just shown is actually a small part of the code for the assembly language program segment for our routine (shown below). The rest is just telling the computer how to talk to BASIC. Now, about the actual routine:

The screen memory for BASIC is set up as 768 character positions (numbered 0 to 767), 24 rows of 32 columns. Each memory cell in screen memory holds a charcter code, which tells the computer which set of defining rows to use to draw the character. The assembly routine given below returns a string variable 8 characters long which contains the printer definition for a character on the screen.

THE ASSEMBLY LANGUAGE FORUM, cont'd

BASIC USAGE:

Before using the PRTCHR routine in your TI BASIC program, you must load it into expansion memory by executing the statements:

CALL INIT to initialize memory
CALL LOAD("DSK1.BSCSUP") to get the support routines
CALL LOAD("DSK1.PRTCHR") to load out routine

All of the required files are supplied on the distribution disk (see below).

The BASIC statement which accesses our routine is:

nnn CALL LINK("PRTCHR",I,A$)

Where I is the character number (0 to 767) to get,
And A$ is the returned string. The string variable can be sent to the printer in graphics mode.

You will be on your own for the BASIC program, depending on your printer. In the listings portion of the article, I have included a version for my Gemini-10X printer. This routine will dump a screen of data, EXACTLY as it appears on the screen (see demo of use with my video version of the codebreaker program). The dump will take about 2 minutes, compared to 15-20 (or more) minutes for a BASIC or Extended BASIC screen dumper.

I realize that this is a pretty heavy first example of assembly language, but the idea is to show what kind of performance inprovement can be gained by even a short routine, like this one. I hope I've whetted your appetite.

For all of these articles, the text of the article, along with source code of assembly and BASIC (if any) programs will be available in machine readable form. No, Virginia, I don't expect you to type them in yourself. Whenever possible, I will try to make extended BASIC versions available on disk, and Mini Memory versions on cassette. Copies will cost you $3.00 if you send me a disk or cassette, $7.50 if you don't supply the disk, $6.00 if you don't supply the cassette. You can get the current month's stuff at the meeting, or send your check to me at:

Vic Kelson
3284 Peggy Lane
Terre Haute, IN 47805
Home phone: (812)466-4031 (after 5pm)

If you want a book on TI assembly language, I suggest:

"Introduction to Assembly Language for the TI Home Computer", by Ralph Molesworth

and

"Fundamentals of TI-99/4A Assembly Language", by M. S. Morley.

My personal preference (for what it's worth) is the second book. It starts out real simply, and has zillions of (well, 47) good examples. The TI Editor/Assembler manual is also a good idea.

Well, that's all for our first meeting. I hope to see at least 4 or 5 (or more) people at the meeting in December. I'll demonstrate the screen dumper, and we'll start from scratch on learning TI-99/4A assembly language. I promise that it'll be real fundamental.

Next month: Another utility to be used with TI BASIC.

LIBRARY BITS

By Dennis Sherfy

The holiday season calls for a review of the Christmas programs in our library. There are five BASIC programs, two of which are duplicates. The first program is listed as MERRY-NEW on the BASIC-4 disk. It is a simple program which displays a Christmas tree on the screen formed by a series of PRINT statements. This technique will make the youngsters in your house instant artists.

MERRYXMAS is also on BASIC-4. It is 50 sectors long and can only be run in BASIC. Because of it's size, you must enter "CALL FILES(1) <ENTER> NEW <ENTER>" prior to loading this program. Otherwise the computer will reserve space for three files and there will be insufficient memory available to run the program. This program creates nice graphics on the screen, including Santa, his sleigh and reindeer, and a decorated tree complete with constantly changing lights. Then, the program plays three Christmas carols--Silent Night, Joy to the World, and We Wish You a Merry Christmas. It repeats these carols until you break the program.

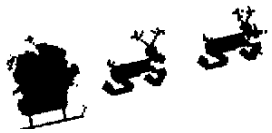
The program M/XMAS on BASIC-9 is essentially the same as MERRYXMAS, but it is 51 sectors long and will not run from disk. I don't know what is contained in the extra sector. To run this program, I had to load it into the console in the same manner as MERRYXMAS, save it onto cassette, then re-load it into the console with the expansion system turned off [using CALL FILES(1)]. I don't know why you would want to do this since the 50 sector version works OK.

SNOOPYNOEL, on BASIC-8, displays an image of Snoopy standing next to a Christmas tree while the program plays "Noel". The lights on the tree change color continuously and the music repeats until you break the program. The last program is SANTA/C, on BASIC-9. This is essentially the graphics from the MERRYXMAS program, but no music.

That's it for Christmas. How about a few of you producing some new holiday programs and sharing them in the library. Do any of you have music files created with the MUSIC MAKER module you would share?

Have a happy and safe holiday.

```
      TITL '* FAST SCREEN DUMP FOR BASIC *'
*
*   BY VICTOR A. KELSON   6 NOVEMBER 1984
*
      DEF  PRTCHR       THE SUBR NAME
*
*   CALLING SEQUENCE:
*         CALL LINK ("PRTCHR",INDX,STR$)
*
*   INDX - (INTEGER) NUMBER OF THE CHARACTER TO CONVERT TO PRINTER IMAGE
*         (0 TO 767) THE BASIC PROGRAM MUST HANDLE THE ROW,COL CONVERSION!
*   STR$ - (STRING) THE PRINTER IMAGE OF THE CHARACTER IN THE INDX POSITION
*
      REF  VMBR,VMBW,VSBR,VSBW
      REF  XMLLNK,NUMREF,STRASG,ERR
*
*   SYSTEM EQUATES
*
FAC   EQU  >834A
*
*   MY BUFFERS
*
CHRBFR BSS  >08          THE CHAR DEFN
STRBFR DATA >0000        THE STRING LENGTH
       BSS  >08          THE PRINTER IMAGE BUFFER
*
*   HERE WE GO! FIRST, GET THE CHAR POSITION
*
PRTCHR CLR  R0           ARRAY COUNT
       LI   R1,1         FIRST ARGUMENT
       BLWP @NUMREF      GET IT
*
       BLWP @XMLLNK      CONVERT TO INTEGER...
       DATA >1200        ...
       MOV  @FAC,R0      IS IT OKAY?? [0,767]
       CI   R0,767
       JH   ERROR        OUT OF RANGE. TELL BASIC
*
*   GET THE TI CHARACTER DEFINITION AND PUT IT IN MY BUFFER
*
       CLR  R1           MAKE THE LSB IN R1 ZERO
       BLWP @VSBR        GET THE CHAR NUMBER
       SRL  R1,8
       SLA  R1,3         MULTIPLY BY 8
*
*   R1 NOW POINTS TO CHAR PATTERN IN VDP RAM
*
       MOV  R1,R0        MOVE TO R0
       LI   R1,CHRBFR    POINTER TO RAM BUFFER
       LI   R2,8         NUMBER OF BYTES
       BLWP @VMBR        GET THE CHAR PATTERN
*
*  NOW,CONVERT THE CHARACTER TO PRINTER IMAGE
*
       LI   R5,STRBFR+1  SET UP THE OUTPUT PTR
       LI   R6,8         AND THE OUTER LOOP COUNT
```



```
*
*   THIS SEGMENT OF CODE SHIFTS THE BYTES IN THE BUFFERTO THE
*   LEFT AND CATCHES THE CARRY BITS IN R0. R0 IS THEN STORED IN
*   OUTPUT BUFFER
*
OUTLP  CLR  R0           CLEAR THE RESULT BYTE
       LI   R1,8         LOOP COUNTER
       LI   R3,CHRBFR    BUFFER PTR
       CLR  R4           FOR INSURANCE
LOOP   SLA  R0,1         SHIFT ANSWER TO LEFT
       MOVB *R3,R4       GET A BYTE FROM THE BUFFER
       SLA  R4,1         SHIFT THE BYTE...
       JNC  SKIPIT       ...DID A "1" FALL OUT?
       INC  R0           YES. PUT IT INTO THE RESULT BYTE
SKIPIT MOVB R4,*R3+      PUT THE SHIFTED BYTE BACK...
       DEC  R1           AND GO BACK TO GET ANOTHER,
       JNE  LOOP         UNTIL THE LOOP COUNTER GOES TO ZERO
*
       SLA  R0,8         PUT OUTPUT BYTE IN MSB OF R0
       MOVB R0,*R5+      PUT THE RESULT BYTE INTO THE BUFFER
       DEC  R6           AND LOOP ON THE NUMBER OF OUTPUT BYTES
       JNE  OUTLP
*
*   NOW, PUT THE OUTPUT BUFFER INTO THE BASIC STRING
*
RETRN  CLR  R0           ARRAY PTR
       LI   R1,2         ARGUMENT NUMBER
       LI   R2,STRBFR    THE STRING BUFFER ADDR
       BLWP @STRASG      THATS ALL.
       RT
*
*   ERROR HANDLING ROUTINE
*
ERROR  LI   R0,>1600     BAD ARG ERROR
       BLWP @ERR
       RT
*
       END
```

```
10000 REM   SCREEN DUMP ROUTINE
10010 REM   VAK, 11/14/84
10020 OPEN #99:"PIO.CR",OUTPUT,DISPLAY
10030 REM  DEMO FOR GEMINI-10X  PRINTER
10040 REM  SET UP CHAR POINTER
10050 PSCD=0
10060 REM  SETUP 1/8" LINEFEEDS
10070 PRINT #1:CHR$(27);CHR$(48)
10080 REM  LOAD THE UTILITIES
10090 CALL INIT
10100 CALL LOAD("DSK1.BSCSUP","DSK1.PRTCHR")
10110 REM  LOOP ON THE 24 ROWS
10120 FOR ISCD=1 TO 24
10130 REM  SET UP GRAPHICS MODE
10140 PRINT #1:CHR$(27);CHR$(75);CHR$(0);CHR$(1);
10150 REM  LOOP ON CHARS IN ROW (32)
10160 REM  GET THE PRINTER VERSION FROM OUR UTILITY
10170 REM  AND PRINT THEM ON THE PRINTER
10180 FOR JSCD=1 TO 32
10190 CALL LINK("PRTCHR",PSCD,ASCD$)
10200 PRINT #1:ASCD$;
10210 PSCD=PSCD+1
10220 NEXT JSCD
10230 REM  SEND A CR-LF TO THE PRINTER
10240 PRINT #1:CHR$(13);CHR$(10);
10250 NEXT ISCD
10260 CLOSE #99
10270 RETURN
```

TIPS FROM THE TIGERCUB #15

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus OH 43213
(614)235-3545

Tigercub Software has over 130 original programs for the TI-99/4A in Basic and Extended Basic, on casette or disk, at only **$3.00** each. A descriptive catalog is available for **$1.00**, which may be deducted from your first order.

I am presently writing Extended Basic versions of many of my programs, enhancing some programs and adding some new ones, and will soon be publishing a new catalog.

No, the Tips are not available by subscription, and I do not have back issues available. However, the entire contents of Tips #1 through Tips #14 are now available on disk, with more added - a full disk of 50 programs, routines and files for only **$15.00** postpaid.

If your local school has purchased TI-99/4A computers for their classrooms, why not let them know that Tigercub has educational programs at a price they can afford?

Since no one else ever reviews any of my software, I'll have to do it myself. **WHITEWATER RUN** is a raft trip down Whitewater Canyon, in 7 levels of difficulty. You must avoid running aground on the green shores, stay away from the black rocks, and do your best to stay out of the whitewater which may conceal other rocks. Every trip is different and none are impossible. The Greenhorn level is a short leisurely ride just for practice. Raftsman is a longer ride which surprises you with a sudden surge of fast water toward the end. Voyageur is a complete change of pace, as your raft leaps through the rapids. The next three levels are such faster, each with an increasing number of hidden rocks in the whitewater. Long Journey takes you through all six levels. Now also available in an Extended Basic version, very fast.

Most of the speed reading programs on the market are worthless, because they flash a phrase from their data file onto the screen and then require you to retype it exactly. Even if the data file of phrases is large, they will soon be recognized from memory rather than from reading. Also, the purpose of speed reading is to quickly grasp the meaning of a text, not the exact wording.

SPEEDER READER avoids these faults by assembling each sentence at random from files of nouns, verbs, adjectives, adverbs and modifiers. The result is an infinite number of different sentences, always grammatical but usually ridiculous. The sentence is flashed on the screen for whatever interval you select; then you are asked any one of several randomly selected questions about it. Where did the bald ballerina kiss the fat judge? If you can keep from laughing too much, your reading speed can be increased greatly.

JUNIOR SPEEDER READER is the same except that the sentences describe the activities of various animals, in simpler words.

Several people have sent me enhancements to my Menu Loader, and I greatly appreciate them. The trouble is, if I incorporated them all the program would take up about 25 disk sectors! So, I have borrowed some ideas, added a few of my own, and here is the result. It will list and load up to 99 programs, stopping at the end of every screenfull or stopping whenever any key is pressed and then offering you the choice of loading, deleting or quitting. It will ask you to verify a deletion by name before deleting it, and will display the name of the program it is loading. It also contains a feature to warn you if you are getting a bad count of disk sectors used - which I find happening more often than you might realize.

```
100 !by A. Kludge/M. Gordon/
T. Boisseau/J. Peterson/etc.
110 CALL CLEAR :: CALL INIT
:: CALL LOAD(8196,63,248)::
CALL LOAD(1163/6,67,85,82,83,
79,82,48,8)
120 CALL LOAD(12288,129,195,
126,165,129,153,102,60)
130 CALL LOAD(12296,2,0,3,24
0,2,1,48,0,2,2,0,8,4,32,32,3
6,4,91):: CALL LINK("CURSOR"
)
140 CALL CLEAR :: CALL SCREE
N(5);: FOR S=1 TO 14 :: CALL
 COLOR(S,7,16):: NEXT S :: C
ALL VCHAR(1,31,1,96):: CALL
COLOR(0,2,16)
150 OPTION BASE 1 :: DIM P$
(99),T$(5)
160 T$(1)="dis/fix" :: T$(2)
="dis/var" :: T$(3)="int/fix
" :: T$(4)="int/var" :: T$(5
)="program"
170 IMAGE ##
180 DISPLAY AT(1,9):"DISKETT
E MENU"
190 ! IF YOU HAVE MORE THAN
ONE DISK DRIVE, DELETE THE !
 IN LINE 200 AND THE FIRST S
TATEMENT IN 210
200 ! DISPLAY AT(12,6):"DISK
? (1-3):" :: ACCEPT AT(12,19
)SIZE(-1)VALIDATE("123"):D$
:: D$="DSK"&D$&"."
210 D$="DSK1." :: OPEN #1:D$
,INPUT ,RELATIVE,INTERNAL ::
 INPUT #1:N$,A,J,K :: DISPLA
Y AT(1,2):SEG$(D$,1,4)&" - D
iskname= "&N$;
220 DISPLAY AT(2,2):"Availab
le=";K;"Used=";J-K:" Prog Fi
lename  Size    Type":"----
---------- ---- -------" ::
T,VT=0 :: TT=J-K
230 FOR X=1 TO 99 :: IF X/20
<>INT(X/20)THEN 260
240 DISPLAY AT(24,1):"Type c
hoice or 99 for more" :: ACC
EPT AT(24,27)VALIDATE(DIGIT)
:K :: IF K=99 THEN 250 :: IF
 K>0 AND K<NN+1 THEN 420 ELS
E 240
250 X=1
260 I=I+1 :: IF I>127 THEN K
=X :: GOTO 360
270 INPUT #1:P$,A,J,B :: NN=
NN+1
280 IF LEN(P$)=0 THEN 320
290 DISPLAY AT(X+4,2):USING
170:NN :: DISPLAY AT(X+4,6):
P$ :: P6$(NN)=P$ :: DISPLAY
AT(X+4,18):USING 170:J :: DI
```

```
SPLAY AT(X+4,22):T$(ABS(A)):
: VT=VT+J
300 CALL KEY(0,KK,ST):: IF S
T=0 THEN 310 :: FLAG=1 :: GO
TO 320
310 NEXT X
320 DISPLAY AT(X+4,1):" " ::
 DISPLAY AT(X+4,2):USING 170
:NN :: DISPLAY AT(X+4,6):"Te
rminate" :: DISPLAY AT(X+5,2
):STR$(NN+1)&" Delete"
330 IF VT=TT OR FLAG=1 THEN
350 :: DISPLAY AT(2,25)SIZE(
4):VT
340 FOR @=1 TO 10 :: DISPLAY
 AT(2,25)SIZE(1):CHR$(30)::
DISPLAY AT(2,25)SIZE(1):" "
:: CALL SOUND(-99,110,0,-4,0
):: NEXT @
350 DISPLAY AT(X+6,1):"    C
hoice?" :: ACCEPT AT(X+6,16)
SIZE(2)VALIDATE(DIGIT):K ::
IF K<>NN AND K<>NN+1 THEN 41
0
360 IF K=NN THEN CALL CLEAR
:: CLOSE #1 :: END
370 DISPLAY AT(X+5,11)SIZE(1
8):" #?" :: ACCEPT AT(X+5,15
)SIZE(2)VALIDATE(DIGIT):KD :
: IF KD<1 OR KD>NN THEN 370
380 DISPLAY AT(X+6,1)SIZE(27
)BEEP:" Verify - Delete ";PG
$(KD);"?" :: DISPLAY AT(X+6,
28)SIZE(1):"Y" :: ACCEPT AT(
X+6,28)SIZE(-1)VALIDATE("YN"
):Q$ :: IF Q$<>"Y" THEN 400
390 DELETE D$&PG$(KD)
400 CLOSE #1 :: CALL VCHAR(1
,3,32,672):: NN=0 :: X=0 ::
GOTO 180
410 IF K<1 OR K>99 OR LEN(PG
$(K))=0 THEN 320
420 CLOSE #1
430 CALL INIT :: CALL PEEK(-
31952,A,B):: CALL PEEK(A*256
+B-65534,A,B):: C=A*256+B-65
534 :: A$=D$&PG$(K):: CALL L
OAD(C,LEN(A$))
440 FOR I=1 TO LEN(A$):: CAL
L LOAD(C+I,ASC(SEG$(A$,I,1))
):: NEXT I :: CALL LOAD(C+I,
0)
450 CALL VCHAR(1,3,32,672)::
 CALL SCREEN(8):: FOR S=0 TO
 14 :: CALL COLOR(S,2,1):: N
EXT S :: DISPLAY AT(12,2):"L
OADING ";A$
460 RUN "DSKX.12345678900"
```

If you don't like my Tigercub cursor, just delete lines 110 (after the CALL CLEAR), 120 and 130. That routine for redefining the cursor has appeared recently in various newsletters without attribution, and I'd like to know who to credit for it. The secret of it is in line 120, where the numbers after 12288 are the decimal equivalents of the hexadecimal numbers (which are the hex equivelant of the binary numbers represented by the off/on pixels) used to redefine a character.

You may have noticed that all programs published in the Tigercub's tips are in 28-column format, just the way they will appear on the screen. And they are printed directly from LISTed actual programs, so that they cannot contain typographical errors - don't you wish the computer magazines did that!? The problem is that when a program listing is merged into the TI-Writer buffer and printed in the formatter mode, the @, &, # and the exponent sign are treated as control characters, and strange things happen!

The following program will convert a program, which has been listed to disk with LIST "DSK1.FILENAME", into a file in 28-column format which can be loaded into TI-Writer, and will optionally substitute the left and right braces, ASCII 124 and the tilde for the @, &, # and the exponent sign, and transliterate them so that they will print correctly in the formatter mode. However, for that very reason this program will not print correctly! When you come to line 280, type DATA shift 2, fctn F, shift 7, fctn 6, shift 6, fctn W, shift 8, fctn A.

```
100 DISPLAY AT(2,4)ERASE ALL
:"28-COLUMN CONVERTER" :: DI
SPLAY AT(5,12):"by Jim Peter
son"
110 DISPLAY AT(7,1):" To con
vert a program, saved":"with
 LIST ""DSK1.FILENAME"",":"i
nto 28-column format which":
"can be merged into the text
"
120 DISPLAY AT(11,1):"buffer
 of TI-Writer."
130 DISPLAY AT(13,1):" Optio
nally with transliter-":"ate
d @, &, # and ^ for cor-":"r
ect printing from formatter"
:"mode."
140 DISPLAY AT(18,1):" Do yo
u want to print the":"file f
rom the":" (E)ditor?":" (F)o
rmatter?"
150 ACCEPT AT(23,1)VALIDATE(
"EF")BEEP:Q$
160 DIM A$(1000):: CALL CLEA
R :: INPUT "What is the FILE
NAME?       DSK1.":FN$ :: FN
$="DSK1."&FN$ :: PRINT : :
170 INPUT "what is the new F
ILENAME?   DSK1.":PN$ :: PN$
="DSK1."&PN$ :: OPEN #1:FN$,
DISPLAY ,VARIABLE 80,INPUT :
: OPEN #2:PN$,DISPLAY ,VARIA
BLE 80,OUTPUT
180 IF Q$="E" THEN 190 :: PR
INT #2:".TL 126;94;" :: PRIN
T #2:".TI 123;64;" :: PRINT
#2:".TL 125;38;" :: PRINT #2
:".TL 124;42;"
190 FOR L=1 TO 1000 :: LINPU
T #1:A$(L):: IF LEN(A$(L-1))
=80 OR LEN(A$(L-1))=160 THEN
 A$(L-1)=A$(L-1)&A$(L):: L=L
-1
200 IF EOF(1)THEN L=L+1 :: G
OTO 220
210 NEXT L
220 FOR J=1 TO L-1 :: S=1
230 FOR T=1 TO 10 :: B$(T)=S
EG$(A$(J),S,28):: IF Q$="E"
THEN 240 :: GOSUB 280
240 S=S+28 :: NEXT T
250 FOR N=1 TO 10 :: IF B$(N
)<>"" THEN PRINT #2:B$(N)
260 NEXT N
270 NEXT J :: CLOSE #2 :: CL
OSE #1 :: END
280 DATA @,@,&,&,^,^,#,#
290 RESTORE 280
300 FOR W=1 TO 4 :: READ CH$
,R$
310 X=POS(B$(T),CH$,1):: IF
X=0 THEN 330
320 B$(T)=SEG$(B$(T),1,X-1)&
R$&SEG$(B$(T),X+1,LEN(B$(T))
):: GOTO 310
330 NEXT W :: RETURN
```

Now, if that's what I give away, isn't it worth a dollar to find out what I'm selling?

Happy hackin'

Jim P.

MicroComputers, Inc.

SALE SALE SALE SALE SALE SALE SALE

TEXAS INSTRUMENTS

INFOCOM SOFTWARE

ENCHANTER	REG. $40. SALE $29.95	STARCROSS	REG. $40. SALE $29.95
INFIDEL	REG. $40. SALE $34.95	SUSPENDED	REG. $45. SALE $34.95
		PLANETFALL	REG. $40. SALE $34.95

ATARISOFT

BUY THREE AT REGULAR PRICE GET ONE FREE
.....FROM THE FOLLOWING TITLES.....

>>>>>>> SAVE $30.00 <<<<<<<

PAC-MAN * DEFENDER * DONKEY KONG * SHAMUS

PROTECTOR II * DIG DUG * PICNIC PARANOIA

MOON PATROL * JUNGLE HUNT * MOON PATROL

MOONBEAM SOFTWARE REG. 19.95 SALE $9.95

ASTROMANIA * CAVEN QUEST * STRIKE FORCE * MOONBEAM EXPRESS
MOONVASION REG. $14.95 SALE $7.95

ROMOX SOFTWARE

PRINCESS AND THE FROG REG. $39.95 SALE $25.95
ANTEATER REG. $34.95 SALE $24.95
TYPO II REG. $39.95 SALE $26.95

PARKER BROTHERS (Q-BERT) REG. $44.95 SALE $27.95

TEXAS INSTUMENTS RECORDERS (WHILE THEY LAST) $35.95

CORCOMP DISK CONTROLLERS $175.00

WE NOW HAVE SUPER SKETCH FOR ONLY $59.95

PROWRITER 8510AP $375.00. GEMINI-10X $ 310.00.

NEC 15LQ AND TTX LETTER QUALITY PRINTER $500.00.

---LIMITED SUPPLY OF TI P-BOXES IN STOCK----

ALL PRICES CASH AND CARRY FOR MEMBERS ONLY

HOOSIER USERS GROUP DIRECTORY

HOOSIER USERS GROUP OFFICERS

President.....Steve Sims 631-7255
Vice-President.....Bill Cagle
Secretary.....Barb Uhrig 357-8268
Treasurer.....Bill Jones

COMMITTEE CHAIRPERSONS

Regional Centers:
South.....Dennis Sherfy 881-5918

Documents.....Don Donlan 882-4544
Membership.....Pam Sims 631-7255
Newsletter.....Pam Sims 631-7255

MEETING LOCATION

Creative Logic
8240 Indy Lane
Indianapolis, IN 46224

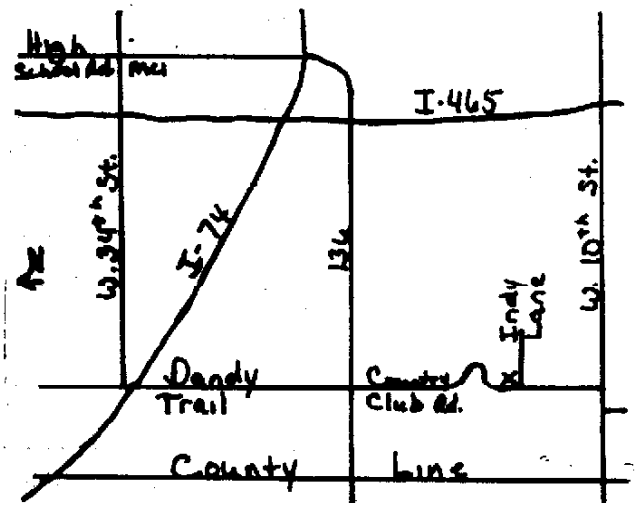
(About 1800 North Country Club Road)

NEWSLETTER EXCHANGE

The Hoosier Users is participating in a Newsletter Exchange program with other TI Users Groups. This offer is made with the understanding that, with proper credit, your Users Group can reprint articles from the Hoosier Users Group Newsletter, and with proper credit, we can reprint articles from other TI Users Groups Newsletters.

PRINTOUTS

Printouts of library listings can be ordered for $.25 & a self addressed envelope with $.37 postage. Printouts of the HUGbbs Reference Guide can be ordered for $.50 and a self addressed envelope with $.20 postage. Please send orders to our P.O. Box. SORRY, PRINTOUTS WILL BE SENT TO ACTIVE MEMBERS ONLY!



HUGbbs INFORMATION

317-631-994A

The HUGbbs operates on a 24 hour basis.

SPONSOR THE HUGbbs: Any member or retail business can sponsor the HUGbbs. For a $5.00 donation, you get 5 (40 column) lines on the Log-On Title Screen for a week (or for a $10.00 donation, you get 10 (40 column) lines) plus a 24 line by 40 character ad in the Sales option of the File Module. To sponsor the HUGbbs, send a check or money order to our P.O. Box (or turn in at our Monthly Meeting) specifying how many weeks (and how many lines) you want to sponsor, your name (or company name), address, phone, what you want to say, and the week (and an alternate week) you want the ad to appear.*

BACK ISSUES

Back Issues purchased at the monthly meeting is $1.00 each. Mail order price is $1.50 per Newsletter (postage included). Orders will be filled within 3 weeks of receipt by the Documents Committee.

ADVERTISING POLICIES

There will be no charge for advertisements submitted to the HUGger Newsletter by members (for private sale only). Format for the advertisements is 45 characters wide by 10 lines long. The Ad should be typed or hand printed exactly how it is to appear in the Newsletter. Deadline for an ad to appear in next month's Newsletter is the 2nd Saturday of the month.*

For companies who wish to advertise in the HUGger Newsletter, our rates are as follows:

Pre-Printed Inserts (one page) $20.00
One Full Page (one sided) Ad: $25.00
One Half Page Ad: $13.00
One Quarter Page Ad: $7.00

All ads must be in a ready to print condition. Advertisements must be in our P.O. Box before the 2nd Saturday of the month to appear in the following month's Newsletter.*

*NOTE: The Officers of the Hoosier Users Group reserve final approval on all advertisements submitted for the HUGger Newsletter and the HUGbbs. The Officers and the Newsletter committee are not responsible for typographical errors due to illegible advertisements. All proceeds are accepted as donations to the Hoosier Users Group.

NOV. 26 1984

HOOSIER USERS GROUP
P. O. BOX 2222
INDIANAPOLIS, IN 46206-2222

ADDRESS CORRECTION REQUESTED

Bulk Rate
U. S. Postage
PAID
Indianapolis, IN
Permit No. 6440

APPLICATION FOR MEMBERSHIP

Below you will find an application for membership to the Hoosier Users Group. Active membership entitles you to the Newsletter, up and download on the HUGbbs, attendance and voting rights at regular club meetings, access to the HUGger Library of Programs, special club activities and special guest speakers for one year. Subscribing members will receive the NEWSLETTER only.

Make check or money order payable to HOOSIER USERS GROUP. Send completed application to:

HOOSIER USERS GROUP
P. O. Box 2222
Indianapolis, IN 46206-2222

please print cut on dotted line

Check One:

Active Member
New: $20_____
Renewal: $15_____

Subscribing Member
New: $10_____
Renewal: $7.5_____

Amount Enclosed _____

#_____ D_____
S_____ O_____

TODAY'S DATE_____

NAME_____

ADDRESS_____APT #_____

CITY_____STATE_____ZIP_____

PHONE (____)____-_____

INTERESTS/
COMMENTS_____

_____