094
8708

50¢

VALLEY OF THE SUN TI 99 USERS GROUP newsletter

VOL.3 AUGUST 8,1987 NO.8

****

CONTENTS

VAST 99 BBS 437-4335

----------AUGUST, 1987----------

VAST INFORMATION

The **VAST USERS' GROUP** is a support group for TI 99 Home Computer and other computer users. Our regular meetings are on the second Saturday of the month. This month's meeting is being held at the Heritage Hotel at 401 N. First Street in downtown Phoenix (corner of Polk and First Streets). We will meet in the Arizona Room. The meetings start at 10:00 AM and continue until 11:00 AM with socializing starting at 9:00 AM. The yearly membership fee is $6.00.

All meetings are open and anyone may attend. Only dues paying members may vote in elections and obtain programs from the Users' Group library.

The current officers are:

President

Vice-President
Stu Olson.....846-7624

Secretary
Bob Nixon.....838-4088

Treasurer
Ike Van Kampen.....934-5164

User Group Librarian
Earl Bonneau.....269-3802

Newsletter Editor/BBS SysOp
Jim Ely.....437-1796

****

A FORTH Tutorial is being conducted by Rene' LeBlanc in this newsletter. It consists of a continuing series of articles relating to his version of FORTH which is available from the User Group Library. For more information, please contact him at (602) 991-1403.

The Users' Group's BBS is now in operation 24 hours a day. Contact it at (602) 437-4335. There is a lot of interesting conversation and information available here so give it a try.

Deadline for submission of articles or advertising for the Newsletter is the last Saturday of every month. Articles may be submitted in any form, however, the preferred method is by phone transfer directly to the Editor.

****

Advertising rates are as follows:

Commercial:

Full Page $10.00
Half Page $ 7.00
Quarter Page $4.00

Personal:

Four lines,
30 Characters/line
$1.00
$.20 per line
over four.

All rates are for **ONE** issue only!

****

Programs are available from the USERS' GROUP LIBRARY at the following rates:

SS/SD Disk $2.00
DS/SD Disk $4.00

If copying of documentation is required, it will be at the rate of $.10 per page. If the User Group supplies the disk, please add $1.00 to the above charges. An exchange program for free programs is also in effect. Please contact the librarian for further information. A complete list of what is in the library is available on 2 disks free of charge if you supply the disks or for $1.00 per disk if the User Group supplies the disks.

----------AUGUST, 1987----------

SECRETARY'S SLATE

M I N U T E S

for

JULY 11, 1987

The July meeting of VAST 99 was held on Saturday July 11, 1987, at the Arizona Title Building at 111 W. Monroe, Phoenix. Stu Olson conducted the meeting as interim president starting at 10:06 a.m.

Minutes from the previous meeting were reviewed but not read. A financial report was given and accepted as well as the minutes.

Tom Moran presented the results of the Select Committee's questionaire that was mailed to all current and past members. At the beginning of the meeting only 27 had been returned. Tom reviewed each question and the percentages of people responding to each question. The final results will be posted to the BBS when completed.

Stu then opened discussion for the revised constitution soliciting changes and/or revisions to same. Motion from the floor to accept constitution as is and not go over item by item. Seconded. Discussion following indicated that majority of those present wanted to review it line by line. Motion failed.

Motion presented and seconded to take constition home and study for a month and review and next meeting. Discussion followed. Motion Failed.

Constitution was then read line by line and changes made to same by approval of majority present. Revised constitution will be placed on BBS when changes made in the document.

Motion and second from the floor to compliment the Select Committee for work done on questionaire and constitution carried.

Stu announced that we would not be able to meet at the Title Building again do to liability problems. Motion and second to appoint a site selection committee carried. Volunteers were Walt, Ike, and Leo.

New constitution requires that a nomination committee be selected to establish a slate of officers for the new association. Motion and second to nominate current board to the new board, carried. Nomination committee to be Walt, Stu, and Dan.

Stu appointed Bob to incorporate new group in accordance with new constitution.

Motion from the floor to elect new officers at todays meeting. Died for a second.

Tom Moran provided an update on the 9640.

Those having not paid dues current were asked to do so.

Meeting adjourned at 11:50 a.m.

Bob Nixon
Secretary

++++++++++++++++++++++++++++++++++++++

EDITOR'S COMMENTS

ON

PAGE 8

++++++++++++++++++++++++++++++++++++++

YOUR PERSONAL

AD

COULD GO HERE!

THE ELEMENTS OF BASIC

by Dave Howell

Courtesy "Erie 99'er Newsletter"

Most microcomputers, including the TI-99/4A, use BASIC (Beginner's Allpurpose Symbolic Instruction Code) as the programming language. Although many different versions of BASIC are used among the various manufacturers of microcomputers, the fundamentals of the BASIC language are the same. The purpose of this column is to describe the elememtary characteristics of BASIC and some of the variations peculiar to the TI-99/4A.

Two modes of programming are available on most microcomputers including the TI: the IMMEDIATE mode, and the PROGRAM mode. The IMMEDIATE mode, sometimes known as the direct or calculator mode, causes a line of code to be executed immediately after entering.

Examples:

```
PRINT "MAKE MY DAY!"
 MAKE MY DAY!

PRINT "32 + 14 + 80"
 32 + 14 + 80

PRINT 32 + 14 + 80
 126
```

If the same 3 lines were placed in the PROGRAM mode, each line must be preceded with its own line number.

Examples:

```
10 PRINT "MAKE MY DAY!"
20 PRINT "32 + 14 + 80"
30 PRINT 32 + 14 + 80
40 PRINT 84
50 END
 RUN
```

The addition of the line numbers causes the computer to accept each line into memory where they wait until the RUN command is entered. Unlike other computers, the END statement is not absolutely required on the TI but it is considered good programming practice to signal the "END" of a BASIC program.

When the above program is RUN, this should be the printout:

```
MAKE MY DAY!
32 + 14 + 80
126
84
```

The PRINT statement causes the contents of the set of quotation marks to be displayed. When working with numerical statements, however, the absence of quotations marks signals the computer to express the numerical statement in its simplest form. No quotes are needed if the numerical statement is already a simple whole number or decimal.

The RUN command causes the computer to execute the program following the order of the line numbers from the smallest to the largest regardless of the order in which they were entered. Line numbers from 1 to 32767 can be used with the TI. However, multiples of 10 or of 100 are used most often to provide greater flexibility in programming.

To make long programs earier to enter, many programmers will use the NUM (or number) command to generate line numbers automatically. NUM 10 or NUM 100 will generate line numbers by 10's or by 100's, respectively. Using NUM by itself generates the line numbers by 10's starting at 100.

----------AUGUST, 1987----------

Computer Tutor Continues

In addition to the RUN command which executes a program, it may be desirable to see a complete listing of the program, line by line. Typing LIST while the above program is in the computer's memory will produce the following printout:

```
10 PRINT "MAKE MY DAY!"
20 PRINT "32 + 14 + 80"
30 PRINT 32 + 14 + 80
40 PRINT 84
50 END
```

Entering "LIST 20" will cause only line 20 to show on the monitor. Entering "LIST 20-40" will show lines 20, 30 and 40 on the screen. "LIST 30-" will display all of the lines from 30 to the end of the program.

Typing "CALL CLEAR" will clear the screen but not the memory. Try it. To prove that the program is still in memory, type LIST. To clear both the screen and the memory, type NEW. It is always good programming practice to type NEW before entering a new program.

HINTS AND TIPS

TMS 9900 ASSEMBLY LANGUAGE TUTORIAL
PART 7
SCREEN BIAS IN BASIC

by Steve Royce

Edit the following code and assemble it using the R option, calling the object code "BIASO".

```
     DEF PR
     REF VSBW        USE VSBW EQU >2024
*                    FOR X-BASIC
PR   LI R0,>02E5     ENTRY POINT IS PR
     LI R1,>4100     >41 IS THE LETTER A
*                    IN ASSEMBLY
     AI R1,>6000     ADD BASIC OFFSET
     BLWP @VSBW      WRITE TO VDP
     RT              BACK TO BASIC
     END
```

Now type in the following BASIC program onto the same disk as your object code, and run it. The letter A will appear in the lower left corner of your screen.

```
100 CALL INIT !LOAD UTIL'S F
ROM E/A CARTRIDGE
110 CALL CLEAR !CLEAR SCREEN
120 CALL LOAD("DSK1.BIASO")
 !LOAD THE OBJ CODE
130 CALL LINK("PR") !LINK TO
 THE ROUTINE
140 GOTO 140 !HOLD SCREEN
```

The very simple example given above helped me to overcome one of the most poorly explained issues in the E/A manual....what does TI mean by the 'SCREEN BIAS OF >60' that must be added to every character that you wish to generate in Assembly but print on the screen from a routine called by BASIC or X-BASIC. A nice simple example like the one above, if it had been in the E/A manual, would have saved me hours.

The reason for the offset is that the pattern descriptor table is located in a different area of VDP RAM when running a pure Assembly program than when running a BASIC program. The difference must be taken into consideration when running Assembly from BASIC.

----------AUGUST, 1987----------

WHEREFORTHS OF FORTH

This is another continuation of our discussion of the DISK-COPY program that was initially presented in WHEREFORTHS #14 (March, 1987). In WHEREFORTHS #17 (June, 1987) I caught up with some errors I had overlooked in the original and some more I inadvertently introduced in the listing that were not in my actual tested code (sorry). In WHEREFORTHS #18 (July, 1987) I described the DISK-COPY word on screen #9. That is the user interface word used to invoke the DISK-COPY function.

I explained what each of its component words does to perform the higher level functions of our program, but did not explain HOW those words are designed. In Forth, we can code our high-level word in terms of yet-to-be defined words that seem to describe the problem steps naturally to us. Then we go to the next level and code those words, and so on. My discussion follows that same sort of sequence. In this issue, I will explain the details of some of the more interesting words that are referenced within the DISK-COPY word.

Since the words I will be discussing range over quite a few Forth screens, I won't repeat those screens in this issue, but ask that you dig out WHEREFORTHS #17 for reference.

GET_#B

This is defined on screen #6. It executes: 0 BLOCK to read the first block (block 0) of the source disk into a buffer. This leaves the address of the buffer on the stack.

Then, 10 + adds 10 to this address. This happens to be the tenth byte of sector zero of the disk, and it and the succeeding byte contain the number of sectors the disk is formatted to have.

Then, @ reads the 16-bit value onto the stack and DUP makes a second copy on the stack.

The CASE statements checks for one of the values 360 (SSSD), 720 (SSDD or DSSD), 1280 (DSDD (Myarc)) or 1440 (DSDD). Note, all of these cases do the same thing and execute the !#B word which divides the value by 4 to convert #sectors to #blocks and stores the result in the variable #B . It is important to understand that in a case statement, any code following the last ENDOF and the ENDCASE statements are treated as an ELSE clause. So in the (unlikely) case someone has a disk with an invalid sector zero where the disk size is not one of the specified values the GET_#B word will DROP the extra copy of the value it read from BLOCK 0 that doesn't match any of the standard CASE values and execute ASK_SIZE which querys the user for the number of disk sectors.

After the ENDCASE, 2 5 GOTOXY places the cursor at col 2 row 5 and then displays a message telling what disk size has been determined.

GET-LO-BUFS

This word executes a DO loop with the loop indices #LB and 1. There is a bit of a trick here. The low RAM buffer space is the five Forth disk buffers. Actually, we need to have one disk buffer as a holding place while transferring buffers to VDP RAM. So I couldn't use buffer 0. Thus we start at 1 instead of 0 for the loop indices.

Within the DO loop the first code is I LO(I). LO(I) is another word that expects a parameter on the stack of the buffer index within the low RAM buffer area on the stack. I is the loop index that starts at 1, then 2, 3, up to #LB-1 as the loop progresses throughout its range. LO(I) multiplies the index by the system

----------AUGUST, 1987----------

WHEREFORTHS OF FORTH CONTINUES

constant B/BUF (bytes per buffer = 1024) and then adds the starting address for that contiguous area of RAM to it (in this case, the system constant FIRST). The result is the absolute address of the I'th buffer in that range.

Following the I LO(I), we see B@ which returns the contents of the Block Pointer BP. This is the system count of which block is being copied. This value starts at 0 and goes all the way up to 359, 719, 1279 or 1439, depending on the disk format. The stack diagram at this point is: (addr blk#).

Next is #B@ which returns the total number of blocks to be copied. The stack diagram is now: (addr blk# lim) where #B@ returns the limit of copying.

Following this is OVER which makes another copy of count on top of the stack leaving the stack diagram: (addr blk# lim blk#). Following this is = which compares the top two values for equality, leaving a boolean flag (truth/false value). This leaves the stack as: (addr blk# f).

The flag will be true if blk#=lim. The IF test consumes the flag and the stack diagram is: (addr count). If the IF test is true, it means all the blocks have been copied and the program must exit. It doesn't need the two parameters on the stack so it executes the 2DROP then LEAVE. LEAVE is a Forth word that changes the loop variable on the Return Stack so the LOOP word will exit the next time it is executed.

Let me explain this end test in more detail, since it is one of the trickier aspects of the COPY-DISK program. Each of the three RAM regions where we want to store disk buffers will be associated with some integral number of buffers that it can hold. These numbers have been determined by the constants #LB, #HB and #VB. There is no guarantee that the total number of blocks to be copied (stored in variable #B) is going to be an even multiple of #LB+#HB+#VB. So, as we execute GET-LO-BUFS, then GET-HI-BUFS and then GET-VDP-BUFS multiple times as we step through the disk being copied, we could in general come to the last block to be copied while part way through using the buffers within LO or HI or VDP BUFS. Each of these words must check for the end before reading each block. If the blk# = limit, the IF test should drop what's on the stack and LEAVE the loop.

Otherwise, the GET-LO-BUFS (in this case) will execute READIT which expects the buffer address and blk# on the stack, and it reads that block of the disk into that buffer area.

You will see that the logic for GET-HI-BUFS is identical to that for GET-LO-BUFS. Only the buffer addresses are different.

GET-VDP-BUFS

This is slightly different, because the VDP RAM cannot be directly addressed. One must first read a disk block into RAM, then write it back out to the VDP. Ironically, the DSR actually reads it into a system buffer in VDP RAM first, then moves it into RAM. It is theoretically possible to take advantage of this by using a very different way of reading the blocks for those destined for the VDP area, but I didn't quite know how to do it, and it seemed more practical to use the BLOCK command which results in bringing the block all the way into the Forth disk buffer area in RAM.

On Screen #8 there are a corresponding set of PUT words which operate the same way as the GET words do, except they execute the WRITEIT word to write a block from its RAM buffer back onto the Target disk.

The GET_#B, GET-LO-BUFS, GET-HI-BUFS, GET-VDP-BUFS, PUT-LO-BUFS, PUT-HI-BUFS and PUT-VDP-BUFS are the main workhorse words of the COPY-DISK program. Next month I'll review some of the other supporting words that provide some of the little input/output niceties of the COPY-DISK program.

Rene' LeBlanc

----------AUGUST, 1987----------

From the Editors Desk

GOOD NEWS... BAD NEWS...

Want to hear the good news first? I thought so.

I sold my house! It only took about a year. That's the good news. The bad news is that there won't be a newsletter for September (unless someone else wants to do one) AND the BBS will be down for at least a week. This is so I can move and get things set up again in the new place. I will be moving to Tempe, so at this meeting we will discuss the status of the BBS phone line etc.

IN THIS ISSUE...

The return of "Computer Tutor" is on page 4. Since Tom Moran thinks he needs a "rest" from doing it, I have picked up a continuing series of articles on "The Elements of Basic" by Dave Howell of the "Erie 99'er User Group". We start right at the beginning with the 2 modes of operation in Basic. Our Assembly Language Tutorial continues on page 5 with a short article on things that change in Assembly when linked from Basic. And Rene' LeBlanc continues his discussion of the FORTH DISK-COPY routine presented a few issues ago. This is on page 6.

There you have this issue...

SOME OTHER NEWS...

I understand that a second version of the MYARC Disk Operating System (MY-DOS) has been uploaded to the pay services. This is certainly good news for Myarc 9640 owners. Speaking of the 9640, J. Peter Hodie of the Boston Computer Society has tried to set the story straight on hardware compatability and the 9640. The TI, CorComp and MYARC disk controllers will all work. The RS232 cards all work and print spooling can be set by the user and is accessed like a normal device, such as PIO. The Horizon RAM disk will work, however, at this time, to boot the system from it, the Horizon EPROM from Genial Computerware must be used. Currently there is support for only 1 Horizon RAM disk. The MYARC 512K card can not be used as is. Myarc can change it for $15 but once changed, you can't use it with the /4A. The speech synthesizer is supported, but you need to buy a special card for the expansion box. The TI 32K and other memory expansion cards, such as the Foundation, will NOT work and neither will the Mechatronics GRAM. The CorComp Triple-Tech card will work but it will eat up about one eighth of your available memory. And finally, the MYARC Winchester Personality card is supported as well as the new MYARC hard drive/floppy controller when it becomes available. If you would like to read the whole article by Mr. Hodie, it is in the Newsletter of the Boston Computer Society for June, 1987.

If you have some interesting news items about your specific computer that you would like to share, just upload a text file (using X-Modem) to the BBS and it will be included here.

ELECTIONS.....

The Nomination Committee has done a terrific job and has come up with a slate of candidates for officers of our group (at least TWO for each office!!!!). If your dues are paid and current, you can vote in today's election. Results will obviously be known at the meeting and will be published in the October Newsletter.

BBS NEWS...

With Summer vacations, BBS usage has slumped a little. I've made a few changes, again, that are obvious if you use the Board much.

You now have the option of replying to the current message you are reading, and if you do reply, the message you are replying to is flagged and a note is printed after the message that there is a reply or replies to that message. This will eventually be changed to the actual message numbers of the replies.

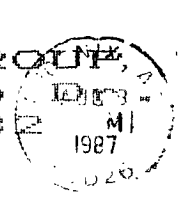
We are still averaging about 80 calls a week, which isn't bad for the Summer vacation period. Keep using the BBS, guys.....

By the way, keep watching the WELCOME message on the BBS for the exact dates it will be down and any new phone number (if there is one).

That's it for this month. See you in October, after I move.

Jim Ely,
Editor,
SysOp

NEWSLETTER
VAST 99 USERS' GROUP
c/o 1425 E. Del Rio Dr.
Tempe, AZ 85282



FIRST CLASS MAIL

TO:

EDMONTON 99ERS COMPUTER SOC.
PO BOX 11983
EDMONTON, ALBERTA
CANADA T5J 3L1

FIRST CLASS MAIL

August, 1987