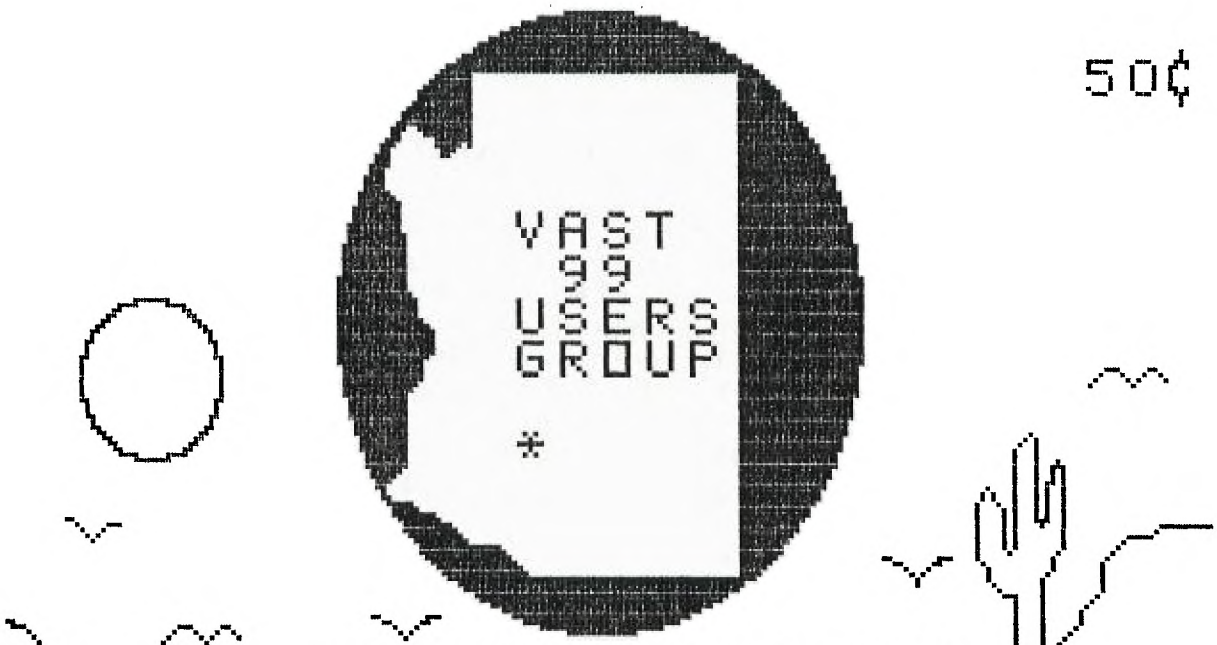
50¢



VALLEY OF THE SUN TI 99 USERS GROUP newsletter

VOL. 3 JULY 11, 1987 NO. 7

****

CONTENTS

VAST 99 BBS 437-4335

----------JULY, 1987----------

VAST 99 INFORMATION

The **VAST 99 USERS' GROUP** is a support group for TI 99 Home Computer users. Our regular meetings are on the second Saturday of the month. This month's meeting is being held in the Arizona Title Building at 111 W. Monroe in downtown Phoenix. It will be in a meeting room on the 10th floor. 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.

****
* *
* Valley of the Sun TI99 Users Group *
* *
****

----------JULY, 1987----------

THE SECRETARYS SHEET

M I N U T E S

for

June 13, 1987

The June meeting of VAST 99 was held on Saturday June 13, 1987, at the Title building on West Monroe in Phoenix. Stu Olson conducted the meeting as Vice President starting at 10:08 a.m.

Stu opened the meeting with the topic that has been on the BBS for the last couple of weeks about a proposal to change the format of the UG to more of a "Generic" group representing several brands of computers. Many of our members have purchased and are using other systems: therefore, we probably need to look at changing the outlook of the group to continue supporting the T.I. and other systems as well.

Stu asked for an open discussion on the topic after indicating that the UG will not attempt to support an IBM BBS. There are already enough of those around.

Comments from members of the UG were generally favorable. A show of hands indicated that 1/3 to 1/2 of the members already had others systems with more on the way. Only one person present expressed a totally negative response to the question.

Recommendations c a m e forward including the following:

1. We should not split up into separate meetings for type of system.

2. A n e w questionaire b e developed to clearly show what type of system and programs each member has.

3. Chairpersons for each type of systems be selected.

4. That more tutorial and training be provided either at the general meeting or other meetings during the month.

Ike Vankampen made several motions that were not recognized until the club voted on the item before it. The group voted to change the format to a generic group.

Ike's motion to form a select committee of four people representing different systems was again presented and seconded.

The committee will review all aspects of the organization, by-laws, and format of the new group. Members volunteering were: Jerry Liddell, Walt Brown, Bill Wedmore, and Tom Moran. The chairman will be chosen from within the group at their first meeting.

Ike also made a motion that an interim president be elected to keep the group functioning. Seconded and carried. Nominations were opened with Stu being nominated. Nominations were closed and Stu elected (Congradtulations (I think)).

A motion from the floor to approve our current meeting place was seconded and approved.

The meeting adjourned at 11:20 a.m.

Bob Nixon
Secretary

/\

SEE PAGE 7

FOR EDITOR'S

COMMENTS

\/

JULY, 1987

WHEREFORTHS OF FORTH

This issue continues the discussion of the DISK-COPY program presented in its (corrected) entirety in WHEREFORTHS #17 (June issue). In WHEREFORTHS #16 (May issue) I explained the purpose of the COPY-DISK program was to copy disks block-by-block using a single drive. The strategy is to minimize the number of swaps required by using all available memory to read the maximum number of blocks before having to swap disks and writing them out. There are three main areas of RAM available: Low RAM, High RAM and VDP RAM. The first thing we had to do was determine how many 1K blocks could fit into each of these regions. Screen #4 defines three constants: #HB, #LB and #VB. Since the DISK-COPY program is itself sharing High RAM, I had to recompute the value for #HB as the last thing on the last screen so that at run time the value would be correct.

In this issue of WHEREFORTHS I will describe the main loop of the DISK-COPY program. It is on screen #9, the last screen of the program. Usually, it is easiest to read Forth programs by going to the very last word defined. It is usually the main user word that one uses to invoke or interact with the program. It is defined in terms of lower-level words that have been added to the Forth language so that reading a Forth program "backwards" gives you a "top-down" viewpoint. Forth programs should be DESIGNED top down, and IMPLEMENTED bottom-up.

```
\ Disk Copy Program SCR#9
 : COPY-DISK TEXT SP! DR0 .SD GET_#B 0 B!
   BEGIN B@ #B@ <
   WHILE CR .RB B@ >R
         GET-LO-BUFS GET-HI-BUFS GET-VDP-BUFS .TD
         CR .WB R> B!
         PUT-LO-BUFS PUT-HI-BUFS PUT-VDP-BUFS .SD
   REPEAT
   CR .COMPL ;


 S0 @ PAD 40 + - B/BUF / ' #HB ! \  Set #HB to final value
```

One invokes the DISK-COPY program (after loading it, of course) by simply typing "DISK-COPY". The program should then prompt you to do whatever is necessary to get your disk copied. Once your Forth program has been loaded, you can (and should) remove your Forth disk unless that is the disk you want to copy. The Forth disk doesn't need to be in the drive anymore once you have loaded the program into memory.

When the user types DISK-COPY, it will begin to execute the following sequence:

TEXT - This puts the screen into TEXT mode.

SP! - This initializes the stack pointer at the beginning since the ACCEPT_AT word may function abnormally if the stack contains extra values left over from previous activity.

DR0 - This selects the first disk drive (Forth numbers beginning with 0).

.SD - This displays a message prompting the user to insert the Source Disk.

GET_#B - This word inspects byte #10 of sector 0 of source disk to see how many blocks it is formatted to contain. This will be the number of blocks to copy.

0 B! - Initialize the Block Pointer variable to zero for our start-up value.

BEGIN B@ #B@ < - This starts a BEGIN-WHILE-REPEAT loop which is the main loop of the COPY-DISK program. It will execute the logic between the WHILE and the REPEAT as long as the end test B@ #B@ < is true. B@ reads the block pointer BP. #B was initialized by the GET_#B word that looked at the source disk to see the "disk size". The "<" test will be true as long as the block pointer variable BP contains a number less than the #B variable that contains the total number of blocks to be copied.

WHILE - begins the WHILE clause.

CR .RB - does a carriage return and displays the message "Reading Block:"

JULY, 1987

WHEREFORTHS OF FORTH CONTINUES

B@ >R - saves the current Block Pointer on the return stack to be used by the "PUT" sequence. In this case, the return stack is just being used for temporary storage. The value must be removed from the return stack before exiting the word.

GET-LO-BUFS - reads as many blocks as there are low RAM buffers to store them.

GET-HI-BUFS - same for the high buffers.

GET-VDP-BUFS - same for the VDP RAM buffers.

.TD - Print the message to switch to the target disk.

CR .WB - Do a carriage return and display the message: "Writing Block: "

R> B! - This gets the saved block pointer off the return stack and reinitializes the actual BP variable with it so the PUT sequence will work on the proper sequence of block numbers. Later we will see that each of the PUT and GET words has to manipulate the block pointer variable to work properly. They also make checks to see if the total number of blocks has been copied so that they can exit when not at a multiple of #LB, #HB or #VB.

PUT-LO-BUFS - writes the low RAM buffers to the target disk.

PUT-HI-BUFS - same for the high RAM buffers.

PUT-VDP-BUFS - same for the VDP RAM buffers.

.SD - Display a message to change to the Source Disk.

REPEAT - When the WHILE test fails, that is, when B@ #B@ < is not true, the total number of blocks have been copied so control jumps AFTER the REPEAT statement.

CR .COMPL - executes a carriage return and displays the message "Disk Copy Complete".

Next month I'll discuss some of the lower level words that are used to support these "top" level functions.

Rene' LeBlanc

HINTS AND TIPS

TMS 9900 ASSEMBLY LANGUAGE TUTORIAL

Part VI

MIXING ASSEMBLY WITH BASIC

by Steve Royce - WNY 99'ERS

If those of us who are 'Assembly Language Snobs' can put our personal biases aside for a few moments, even we can admit that many if not most things can be adequately handled by TI BASIC or Extended BASIC. Only a few things slow us down to the point where we resort to twiddling our thumbs as we wait for our BASIC program to get to the next step. Alphabetic sorting, for example, is best handled in Assembly, but input of the data to be sorted from keyboard or disk is, let's face it, quite a bit easier in a BASIC language. What we need is a method to pass the data back and forth so we can use the best features of both languages.

Fortunately for us, TI TMS9900 Assembly Language provides two sets of utilities: STRASG and STRREF for passing string variables and NUMASG and NUMREF for passing numerical data. In this article, we present a simple example to show the STRing util-

HINTS AND TIPS CONT.

ties. Note that the routines presented are probably not of great value in their present form, but should at least reveal the concepts necessary to expand them into more general and useful routines for your use.

Most of the documentation necessary for their use is presented in the REM's and comments, but a little explanation may be valuable. First, the BASALS routine passes strings from BASIC to Assembly Language. The ALBASS routine is the reverse routine. Second, I had a hard time understanding what TI's E/A manual meant by 'argument number.' To put this simply, in a statement like 'CALL LINK('ALBAS$',A$,B$,C$), A$ is the first argument, B$ is the second, and so on. The third point I would like to explain is that a byte must first be moved into the BUFFER which is to hold the string passed from BASIC to Assembly. This byte defines the maximum length of the string to be moved. If a longer string is attempted, it will be truncated.

Enter the following source code using the Editor/ Assembler, assemble it as 'DSK1.OBJECT' using the 'R' option. Don't use the 'C' option as the Extended BASIC loader can't handle compressed object code.

```
       DEF BASALS,ASBASS
STRREF EQU >2014
STRASG EQU >2010
HEX20  BYTE >20
       EVEN
BUFFER BSS >22
*
* Routine to move string Basic to CPU buffer
*
BASALS MOVB @HEX20,@BUFFER -Allots >20 bytes
                            to string
       LI R0,0             -Not an array
       LI R1,1             -Pass the first
                            argument
       LI R2,BUFFER        -Where to put it
                            in CPU
       BLWP @STRREF
       RT                  -Back to BASIC
*
* Routine to pass from CPU back to BASIC
*
ALBASS LI R0,0             -Not an array
       LI R1,2             -Pass to second
                            argument
       LI R2,BUFFER        -What to pass
                            back
       BLWP @STRASG
       RT                  -Back to BASIC
       END
```

Note that the above routine is pretty specific, in that it can only pass the first argument from BASIC and only pass back to the second argument from the CPU. However, generalization of the routine to make it pass any argument is not difficult.

Now the Extended BASIC calling program:

```
100 CALL INIT
110 CALL LOAD("DSK1.OBJECT")
120 A$="ONE" :: B$="TWO" ::
C$=" THREE"
130 D$="FOUR" :: E$="FIVE"
140 REM MOVE D$ TO BUFFER
150 CALL LINK("BASALS",D$)
160 REM MOVE BUFFER TO B$
170 REM NOTE THAT THE ASBASS
 ROUTINE AS WRITTEN WILL PAS
S
180 REM THE BUFFER BACK TO T
HE SECOND ARGUMENT, SO B$
190 REM MUST BE THE SECOND A
RGUMENT OF THE LINK
200 CALL LINK("ALBASS",A$,B$
)
210 PRINT A$:B$:C$:D$:E$ ::
END
```

The print line should now show that D$ (FOUR) has been moved to B$. Not too impressive an example, but you get the point.

If you want to use the routine from BASIC with the E/A module rather than from Ex-BASIC, simply modify the multiple statement lines and add the following line:

```
105 CALL LOAD("DSK1.BSCSUP")
```

and load the BASIC support utilities from E/A disk A. Also, change the EQU's in the assembly language source code to:

```
REF STRREF,STRASG
```

Keep up the efforts in Assembly. My next article will involve writing to the screen from Assembly, then returning to BASIC. This should help explain that menacing thing TI calls a 'screen bias of >60'. Till next time.

----------JULY, 1987----------

From the Editors Desk

Well, here it is July already. And since it started, I don't think the temperatures have b e e n below 100 degrees during the day and mostly in the 105-110 degree range. Summer sure is here!!

IN THIS ISSUE...

This month's issue is a little shorter as I have been busy with other projects.

On page 4 is Rene's WHEREFORTHS continuing with the DISK-COPY program description written in Forth. Part 6 of the Assembly Language Tutorials starts on page 5. I must explain something to those of you who are following this TUT.

I got the Assembly TUTS from one of the other user groups as a 10 part series. I do have 10 articles for publication but it seems, in reading through them, that some are missing. What was described in last month's issue as continuing this month is not what we have here (we were supposed to be writing files to disk). I will try to contact Steve Royce who wrote the articles and see if I can get what seems to be the missing articles. Sorry about that....

That's all I have for this month. Maybe someone else in the group will submit an article for publication next month (WISHFUL THINKING.....).

IN OTHER THINGS...

I understand that Dan Shell got 2 (not 1, but 2!!!!) of the NEW Myarc 9640 Computer and has already sold one of them to Jerry Liddell! Do we have one this month for a demo???

In reading MICROpendium for July, the DOS (Disk Operating System) for the new Myarc is now complete (they got it just as the July issue had gone to press). This thing is really going to happen!!!

I got a package from INNOVATIVE PROGRAMMING of Rohnert Park, CA. This must be a new company as I have not heard of them before. The package included a disk with a few utility programs on it and some descriptive info on the products they carry. The disk will be turned over to the librarian and the literature is available to those wishing to view it.

There is a new DOS package out now for the TI-99/4a with commands similar to MS-DOS. This should prove rather interesting and exciting. It is written by Monty Schmidt and is available from Ryte Data for $19.95. Sounds like a great deal.

EDITORIALIZING....

Last month we elected a committee to handle the rewritting of the group constitution. This month, that group will present it's rewrite for group approval. Some of you have expressed concern that in all the shuffle the TI will be eventually over-looked in favor of "other" computers. Do you know what it is going to take for this NOT to happen? Well if you read the new constitution, you will see that the only computer specifically mentioned is the TI. However, for it to stay that way, we need strong LEADERSHIP in that direction. A lot of you want and only need TI information. We need people to get that information and present that information to the group. We need programs set up at EVERY meeting specifically for the TI. Did YOU buy a new piece of hardware or software for your computer? WE would like to see a demo of it. Are YOU pretty good at programming in any particular language? WE could use a workshop in it. I'm not talking about the guys who have been doing it all along. I'm talking about YOU! WE need YOU! Please take an active role in keeping the TI THE major part of this group.

Jim Ely,
Editor,
BBS 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

July, 1987