

TIOSQA

TI PUGET SOUND 99ERS

P.O. BOX 6073
LYNNWOOD, WA 98036

BACK RATE
U.S. POSTAGE
PAID
Lynnwood, WA 98036

Ø35 8703
Puget Sound

MARCH 1987

Vol. 6 No. 03

MEMBERSHIP \$15.00 PER YEAR

NEWSLETTER PRICE (FOR NON-MEMBERS) \$1.50 PER ISSUE

OFFICERS

PRESIDENT :	CHUCK WYNNE	745-3249	VICE PRESIDENT:	GENE CHRISTIANSEN (Call Chuck Wynne)
SECRETARY :	DENNIS WOOD	641-0298	TREASURER :	DAVE POWELL 631-7794
NEWSLETTER:	TOM WYNNE	745-3249	CLERK :	JOHN ENGLAND 672-7793

NEXT MEETING

DATE : THURSDAY, April 16, 1987 TIME: 7:00 PM
PLACE: Kirkland Public Library, 406 Kirkland Ave, Kirkland

- AGENDA:
- * Old business
 - * SPAD VIII flight simulator demo
 - * Demonstration/Lecture
 - * T199/4A new news and info
 - * Library updates-John Ueland
 - * Question and Answer session

MAY MEETING

DATE : Thursday, May 21, 1987
TIME : 7:00 PM
PLACE: Kirkland Public Library
406 Kirkland Ave, Kirkland

JUNE MEETING

DATE : Thursday, June 18, 1987
TIME : 7:00 PM
PLACE: Kirkland Public Library
406 Kirkland Ave, Kirkland

IN THIS NEWSLETTER

The President Says.	3
W A M	4
TI99/4A News	
Queen Anne Computer Shoppe new location.	5
New flight simulator available.	6
<u>Reviews, Comments, Tricks, Hints</u>	
Disk Controllers - an Addendum.	6
Multiplan Machinations.	7
<u>Programming the TI99/4A</u>	
BASIC:	
Extended BASIC tips, tricks, and hints.	12
Extended Basic Subroutines.	13
ASSEMBLY LANGUAGE:	
LEARNING TI99/4A ASSEMBLY - PART 2.	15
FORTH:	
TI FORTH PART 2	18

REGIONAL REPRESENTATIVE MEMBERS

<u>TITLE</u>	<u>PHONE NUMBER</u>
PUGET SOUND 99ERS PPS-SEATTLE	784-4142
TIBBS NW-ED DURFFEE-BELLEVUE	641-5884
QUEEN ANNE COMPUTER SHOPPE	361-0895
NEW WORLD BBS	365-6938
BAINBRIDGE ISLAND	1-598-3228
WLETRAVEL	742-9034
SHUTTLE	885-INFO
TROTHGUARD-BURIEN	824-6757
RADIO 1	878-0158
Genie	1-800-638-8369

THE PRESIDENT SAYS

By Chuck Wynn

A tip of the hat to all concerned for the job of getting Ralph Fowlers board to work with our General Datacom modem. Keith got a more sensitive opto-coupler for the speed sensing lead (pin 12), and it worked. Now we can send Ralph the money for the program and the support. We have the Triple-Tech card working for real time and date now. I would also like to thank Chris George for his work on the board in the past. We aren't passing this work lightly. I sure could use his support on the new software for improvements, etc. How about it Chris?

Ryte Data says it has a chip to replace one in the TI disk controller that will allow DSDD and or Quad to run off of that card. Haven't seen it yet, but believe me I'm looking too.

Hey, I am too used to using a modem. In writing this article, I started to use a CTRL H to backspace and found out a couple of things about TI-Writer. A CTRL H goes to the start of previous paragraph. A CTRL J goes to start of next paragraph. Try it!

Here's an old trick that maybe got lost. Plug in the TEII module. Load a program in in basic. When loaded, type in LIST"SPEECH":110 (any line number you wish to check). This is good to check data statements in your programs for errors. It won't be of any good for the XB statements, but is good for easy chair listening of other ones.

From Byte Magazine April 1987 Page 52. Two words of caution in buying a quad system. You need to use quad density disks. You also won't be able to write disks for use in 360K byte drives even if you format them for 360K. The problems come from the size of the read/write head (it's much narrower) and the strength of the magnetic field (it's a lot weaker) in the quad-density drives compared to standard drives. If a quad density drive writes or erases, it is a much narrower path and the DSDD drive will read a wide path. Thus, some data can look like garbage. Reading anything is just fine with the quad drive. Just use it for writing or erasing quad disks only!

Just imagine this bit of information. A disk drive rotates at 300 RPM. When the drive reads a disk, a minimum of 250,000 magnetic bits pass under the head each second. Ever heard of interlace? Well it seems the computer can't move memory and read consecutive sectors on a disk. So they didn't put the data to be read in consecutive order. Instead, they spaced it out by using a given number of sectors in between. These aren't lost, they just are used elsewhere. The head reads a proper sector, then must read

the next consecutive sector which is several sectors away on the disk, but the read head by this time is conveniently over that one. This improves disk retrieval time as it can get the job done in one revolution instead of having to wait until the disk turns again. Interlace is important for speeding up the job. It is fixed and cannot be changed on the TI disk controller. It was made this way so as to be compatible with just about any drive, slow or not.

R-A-M

By GENE CHRISTIANSEN

It has occurred to me that not a great many read or have read this column. This is because, while I have tried to be a bit outspoken on certain issues, I have yet to hear anyone praise or condemn my efforts. I HAVE heard that this newsletter is widely read and appreciated, though from the lack of input by our membership would indicate otherwise. I suppose that most are content to just be "hangers-on" in lieu of full participation in the organization. Let me state that this attitude only promotes a user's group that is "hanging-on". People come up to me at meetings and complain "Where's my newsletter?". My response to these vocal do-nothings is "Where's your input TO the newsletter?".

It is pretty easy to lodge complaints (real or imagined) while not submitting articles or programs for publication. It is pretty easy to denounce the work of others while not even VOTING for different leadership. (GOD FORBID THAT WE SHOULD CONSIDER RUNNING FOR OFFICE OUR SELVES!!!!)

It is unfortunate that even these comments are not new. It seems that no one really cares. Could it be that the membership in this area has ACCEPTED the demise of the 99/4A and is now just waiting for the various clubs themselves to fold? That is what will happen without member support!!!

To further illustrate my point, some time ago I started a promotion. Its object was to bolster our membership and, at the same time, provide support to many who did not belong to the group (or even know of its existence!). The idea was simple enough: BE RESPONSIBLE FOR TWO NEW MEMBERS AND RECEIVE A BOX OF TEN (10) DISKETTES. When I last checked with the club librarian, we had NOT GIVEN A SINGLE BOX AWAY!

It was mentioned at the last meeting that the newsletter was late because of the difficulty in find material to fill its pages. (This came after many howls from those in attendance for not having their copy already.) There are many, like former club secretary George Sjaastad, who criticize long and loud about the group but rarely, if

ever, contribute anything. I single George out because during the year that he held office, the minutes of our meetings appeared only twice - and those were written by Chuck Wynne!!!

However, George is quick to quote the by-laws, etc. to make sure that he gets his copy!!

Want to help? See Chuck or Tom Wynne at a meeting and tell them that you want to lend a hand. One of the hardest tasks is to review the newsletters from other groups to find items of interest to our membership. There is a lot of information going around in the TI 99/4A world and those who do not have access to other publications (or subscribe to the Source or Compuserve) miss about 80% of what is going on.

It is good to see Regena in print once again. Since the first of the year she has been gracing the pages of Micropendium. She still provides those wonderfully educational programs in BASIC and Extended BASIC as well as tips on the TI in general. Check it out!!!

A new flight simulator has been released by Not-Polyoptics. Rated very high by Micropendium, the program retails for \$29.95.

A company named DIJIT Systems has released an RGB conversion kit for the 99/4A. This too was rated very high by the staff at Micropendium. The package retails for about \$19.95.

Well, that is about all for now. Drop me a line if I have stirred you. I don't care what you say as long as you say something!!! B-M out!!

TI 99/4A NEWS

Queen Anne Computer Shoppe new location

Queen Anne Computer Shoppe has moved off of the hill. It is now on Roosevelt Way and 62nd street. A marked improvement in space and design. Come see Barbara there and support her new business location. She has offered to allow us to have our meetings there. This would mean a place to see and review new software, hardware, new revelations in the TI world, and best of all, the TI computers (9640 too) would already be set up for us to use. Comments?

New flight simulator available

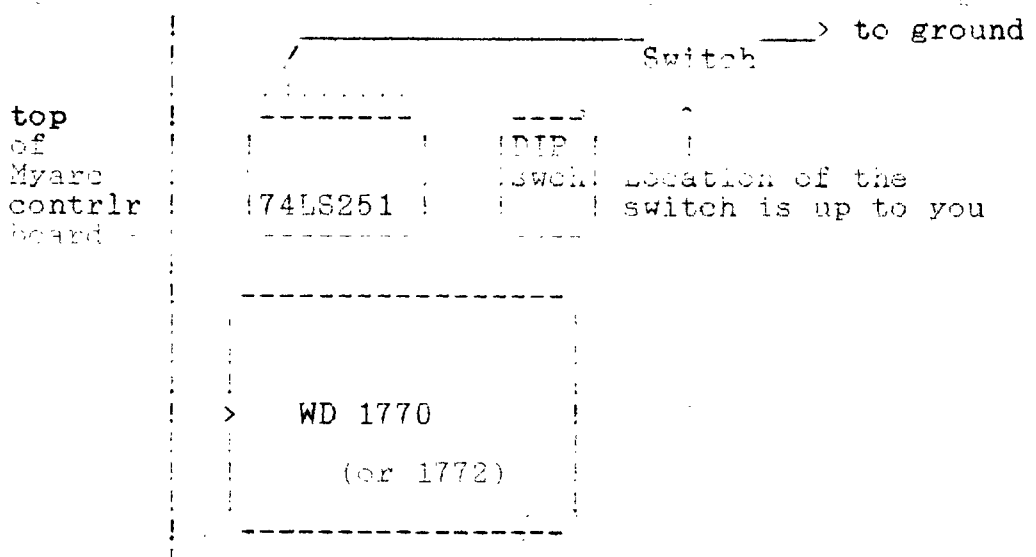
The Spad XIII flight simulator by Not-polyoptics is the latest of the flight simulators available for the TI99/4A. This flight simulator enables you to fly a World War I plane and with realistic graphics! MICROpendium reviewed this in depth in the March 1987 issue. The Spad XIII flight simulator is available at the Queen Anne Computer Shoppe.

Disk Controllers -- an Addendum

Jerry Coffey [74716.3525] - March, 1987

I mentioned a "turbo" modification to lock out the "read after write" (write verify) routine usually performed by the controller. Here are the details:

Find the 74LS251 chip at the top center of the controller board, above the DIP switches and beside the large FDC chip (marked WD1770). Solder a wire from the number 2 pin of the 74LS251 through a switch to ground (e.g. the wide trace of the DIP switches or any trace connected to that wide trace). It looks about like this from the bottom (non- component side) of the board.



As always you proceed at your own risk. (One person has told me this did not work on his 40 track system, but I haven't verified that.) You can tell it is working if your controller writes as fast as it reads (normally the write takes twice as long).

Since I wrote the article on disk controllers, I have discovered some surprising facts about my own system. All of the Myarc timings in the article were done on an 80 track system with the fast WD1772 controller chip (stepping at

2ms). With some help from Paul Charlton and Richard Roseen, I recently customized Paul's Eprom to step at 3ms using WD1772. (Richard's drives were making errors at the faster speed.) I used a Mechatronics Eprom programmer to download the Eprom code to disk, changed the FDC commands with a sector editor, and wrote the altered code back to a fresh Eprom. The process is simple (and cheap) once you decide what code you need in the Eprom.

The slower step speed made it possible to notice some slight differences in the performance of the WD1772. The first thing I noticed was that interlace 4 on 18 sector tracks was no longer smooth -- it was missing the first sector after a track seek and forcing an extra revolution of the disk. This was the first clear indication of how close this format is to the "ragged edge". The reaction to the small change in step speed implies that this interlace comes within 5% of the minimum time required to step and settle the head. Thus the likelihood of read/write errors is relatively high with this interlace. It will occasionally detect the sector ID and begin to read or write before the head has completely settled. This interlace should definitely be avoided -- 18/3 is both faster and more reliable.

The Eprom modification itself was an interesting experience. I patched the new FDC commands into some unused text bytes and patched addresses into the code to point to the new locations. The Mechatronics Eprom Programmer is an excellent piece of equipment. It will burn (program) a 2764 (8k) in about 90 seconds using the fast algorithm. I have talked to Jim Horn and Jeff Guide about offering an Eprom service to the customers of Disk Only Software. There are many possibilities this technique opens up. There is the 80 track modification for the TI controller worked out by Bob Cooper. And many Myarc owners are still using old Eproms that have never been upgraded (though this situation has improved since Lou Phillips increased his production capacity). The fix we developed for Richard's controller can provide the optimum step speed (3ms or 5ms) for different disk drives using the WD1772 FDC chip. And any enterprising programmer can get his tailor-made code installed in nonvolatile memory.

MULTIPLAN Machinations

-----in the Harm's way.

More on Basic programs that create Symbolic Link (SYLK) files to load data into Microsoft's Multiplan.

The first article showed how to create a Multiplan (1) SYLK file for one cell. It only described a few of the many

SYLK symbols. See pages 205-208 of the Multiplan Manual for a complete list.

The most significant change in this month's program is that it can create a file with many cells -- a couple of columns of data.

After using this program, you could load the file into Multiplan and save it as a Normal file. Then you could eXternal Copy it or parts of it into another spreadsheet that has formulas and other data. You might load a month of information into a sheet that has many months of data plus formulas for calculating Year-to-Date, Average, Units per Time Period, etc.

As I mentioned last month, the Basic program must write a Display Fixed 128 file. You must change the File Descriptor Record (FDR) to make it look like the file is Internal Fixed 128. Otherwise Multiplan cannot read it. The only trick is to find the FDR!

This is a very sad deal, but it is very easy to change it with a good sector editor such as Miller Graphic's ADVANCED DIAGNOSTICS. Change byte 12 of the FDR for your SYLK file from Hex 0002 to Hex 0202. The FDR's are in Sectors 2 thru 32.

Another way is to use Barry Traver's fine program RAW (Read And Write) which is written in Assembly language. It will make the change right out of your Extended Basic program. It was published in the Super 99 Monthly by R.M. Mitchell in the summer of 1985.

Each SYLK record (remember - files are made up of records) can include cell content (text, numeric value or formula), row and/or column numbers and many other symbols for you to describe windows, sheet bounds, formatting of sheet and individual cells, sheet links and more. In fact, you can include just about everything except Multiplan commands such as COPY and DELETE.

Your cell content data can be split into more than one record. You just keep on creating 128 character records of SYLK symbols and data until you run out of information that you want put into the SYLK file. Then you fill the last record in the SYLK file with nulls <CHR\$(0)> so that it is 128 characters long.

Last month's program had 27 nulls which made it a 73 byte record. I couldn't get it to load using a larger number of nulls (like 123). R. Mitchell's program as published in the May, June and August, 1985 SUPER 99 MONTHLY had the key.

You continue building a string (numerics are included via STR\$(xx)), using the & to concatenate each new symbol or data item onto the string as you go UNTIL the LENGTH of the string exceeds 128. Then you print the first 128 characters

of the string into your disk SYLK file. Right after printing you have any characters in your string beyond 128

Then you move the remaining characters back into your main string and continue building until it has more than 128 characters again and you do another print to disk.

There are two types of SYLK symbols: Record Descriptors (RTD) of 1 or 2 characters and Field Descriptors (FTD) and are the same as in the Multiplan manual). CR/LF's are used to separate SYLK record types (careful - a Record Type or RTD is a SYLK symbol while a record is part of a file - records contain record types).

Lets look at a sample: "C;K". This is a RTD C which means that it is a data point. The FTD ;K means that the value of the data point follows. Once you set a row or column number with the RTD C and FTD of say ;Y (for row), all the remaining data points (;K or ;E) are put at that row. You only need give another "C;X" or "C;Y" when changing the row (;Y) or column (;X) of the data point.

One of the fastest ways to learn the correct formatting of SYLK records is to enter some stuff into Multiplan and save the sheet in Symbolic format. Then you can look at the file with a sector editor like ADVANCED DIAGNOSTICS.

The following program creates a SYLK file of 2 columns

since a name of a cell or a range of cells can be used in a formula (to wit) SALES-COST (SALES minus COST). You would have NAMED some cells SALES and some others COST, so the cell with this formula yields the profit. It could be named PROFIT.

A normal formula might look like A100-B100. For a bunch of stuff you might use SUM(R1:40C3)+SUM(R90C4:50C5). Relative references (relative to the cell that has the formula) as in my program look like SUM(R[-5]C:R[-2]C).

The whole sheet looks like this when loaded into Multiplan:

column	1	2
1	13	5
2	1	70
r	30	109
c	400	75
w	3	-----
e		138

```

A 1 C 1 1 . A 1 U 3 T C ,
Y 2 ; K 1 * * C ; Y 3 ; K 3
3 * * C ; Y 4 ; K 4 0 0 * *
C . Y 2 2 2 1 1 1 1
. . . . .
- - * * w , 0 1 , A 1 1
* * E * * * * * * * * * *

```

Line 128 Character SYLK record ends after the fourth character in the 15th line. This is also the end of the

R. Mitchell's program reads a Display Variable 80 file and writes the data into a SYLK file that Multiplan can read. It has one word processor line on each row with 5

lines. The record 25 columns and 40 rows of formulas, data and labels (text) is about all that the 4A's RAM can hold.

The program I wrote, which is based on the following follows. Write Bytemaster Computer Services, 171 Mustang St., Sulphur, LA 70663 for a subscription to the "The Smart Programmer" and back issues, even on disk. These comments may help you understand it.

Line 300 adds sheet identification, windows, sheet and bounds.

Line 310 adds the name you inputted in line 280 and the range for that name (R1:6C2).

Line 320 adds the formula for R6C2.

Line 400 adds the cell content of formulas have taken from the array AYT.

Lines 370, 450 and 470 work together to ensure that each

Now you can load your data into Multiplan to suit your needs. Instead of DATA statements you can use a file INPUT statement instead of the READ statement in line 210.

*** E X P L O R E *** by Bill Harms

the August, 1986 issue. It includes instructions for loading your SYLK file into Multiplan. This article was edited for the ROM by Jim Swedlow. ED]

```
100 SYLK 11/80
110 DISPLAY AT(1,5)ERASE ALL:"A SMALL SPREADSHEET"
120
130 OPTION BASE 1
140 DIM AMT(4,2) array for input data:*****
150
160
170
180
190
200
210
220
230
240
250
260 ACCEPT AT(7,4)BEEP SIZE(-15):FILE2$::IF FILE2$=""
    THEN 260
270 OPEN #2:FILE2$,DISPLAY ,FIXED 128, OUTPUT
280 DISPLAY AT(9,1):"Enter DESIRED name for 2nd column. "
    :: ACCEPT AT(10,1 0)SIZE(10)VALIDATE(UALPHA):NAME$ ::
    IF NAME$="" THEN 260
290 NAME$=NAME$+CHR$(10)
300 T$="ID;PMP"&R$&"F;W1 2 6"&R$&"F;DG2G8"&R$&"B;Y"&
    R$&"A-D";X=STR$(1)
310 T$=T$&"N;N"&NAME$&" ;ER1:6L2"&R$
320 T$=T$&"C;Y6;X2"&" ;ESUM(R[-5]C:R[-2]C)"&R$
330 !
340 FOR COL=1 TO 2
350 T$=T$&"C;X"&STR$(COL)
360 FOR ROW=1 TO A-1
370 IF LEN(T$)>128 THEN CALL WRITE(T$,T1$)::T$=T1$
380 T$=T$&"C"
390 T$=T$&" ;Y"&STR$(ROW)
```

```

400 T$=T$&" ;K"&STR$(AMT(ROW,COL))&R$
410 NEXT ROW

440 T$=T$&L;Y &STR$(ROW)& ;N VCHAR$(34)& TTTTT &
CHR$(34)&R$
450 IF LEN(T$)>128 THEN CALL WRITE(T$,T1$)::T$=T1$
460 T$=T$&"W;N1;A1 1"&R$&"E"&R$
470 IF LEN(T$)>128 THEN CALL WRITE(T$,T1$)::T$=T1$
480 PRINT #2:T$&RPT$(CHR$(0),128-LEN(T$))
490 CLOSE #2
500 END
510 SUB WRITE(T$,T1$)
520 PRINT #2:SEG$(T$,1,128)
530 T1$=SEG$(T$,129,LEN(T$)-128)
540 SUBEND

```

Extended BASIC tips, tricks, and hints

The Margin Problem
 Tom Fyfe

Many people have wondered why TI let us only use 28 columns for printing or inputting text through TI BASIC or TI Extended BASIC. Well some television sets would not be able to display things on the far right or left. Some people have the problem of not being able to see the cursor or the "=" is cut off the left side. Now with many using the TI-84C Plus, the problem is solved. The TI-84C Plus can display the far left and far right two columns. In many cases, 28 columns is not a very large area to work with. Especially when using the ACCEPT AT statement, for it can only use one line at a time. Many things have been tried, the most successful, using an Assembly Language CALL LINK to solve this problem. Others have gone to a completely different Extended BASIC. There is no way of changing this problem of 28 columns without having to either get a new enhanced BASIC or using the GRAM Kracker to change from 28 columns to 32 columns. This is because the ACCEPT AT, PRINT, and DISPLAY AT are written in ROM and BIOS (Read Only Memory) so they are permanent.

I have stumbled upon a bug (or feature) in the ACCEPT AT statement that explains the way it will not allow you to use the right and left columns of the screen. The following small programs are examples that you can play around with and change. The first is the program that I tried when I found the bug.

```

10 CALL VCHAR(1,1,31,768)

```

```
10 ACCEPT AT(1,1)SIZE(-28)
20 GOTO 20
```

This program first fills the entire screen with character 31, the edge character. Then line 20 accepts a string from the keyboard from row 12. The SIZE(-28) reads whatever is on line 12 and allows you to change it. When you run the program, start typing. Notice that every character you type is separated by 4 spaces! And also that you can type forever, and that some of the letters you type are in the margins of the screen! (If you keep typing until the top has scrolled off the screen and press ENTER, you get pretty colors and patterns that keep changing with every key press.) Well this is not very usefull but it shows that whenever your cursor is next to a character 31, it automatically skips 4 spaces. This is because character 31 is supposed to be on the left 2 columns and the right 2 columns. When you type to the right side of the screen, it skips 4 characters to get to the left side, skipping over the margins. The next program is a revision that almost works. It unfortunately messes up the first line but all subsequent lines are typed in the margins and the result is what you typed.

```
10 CALL HCHAR(1,30,31)
20 ACCEPT AT(1,1)SIZE(-28):A$
30 GOTO 20
```

Again you can experiment and try different combinations of line 20.

Extended BASIC subroutines

By Tom Wynne

I have been working on various programs for people that help them with their business. One is Fred Pattison. He runs Pattison's portrait studio and a barber shop in Auburn, Washington. And he runs them both with his TI99/4A computer. I have, and still am working on, an invoice, inventory, and price list package that he can use for running his studio and keep track of various operations of his business. I have written these programs all in Extended BASIC. Don't believe the people that say BASIC is a language for beginners and is only used for educational purposes. BASIC is the most widely used language for many applications. The reason is that it is easy to change, no need for compilation (optional) and is relatively easy to program and understand. These are some reasons why many people prefer programming in BASIC rather than other

languages.

In the programs that I have written for people, I have had to make some subroutines that I commonly use. This is good programming practice to create a subroutine for a particular program and save it away so you can use it with any other programs that you write in the future.

Here are some of the subroutines I have written for some of the work I have done:

The first is a routine that will convert a string to upper case. This is mostly used for searches in databases. If someone enters "Hello" and the data base has stored "HELLO" the UPPER function would be used to convert both the search string and the data in the database to uppercase to compare them to find a match. If it did not do this, "Hello" and "HELLO" would be considered two different words and it would not find it.

```

7000 SUB UPPER(A$)
9010 FOR I=1 TO LEN(A$)
9020 T=ASC(SEG$(A$,I,1))::IF T>96 AND T<123 THEN T=T-32
9030 B$=B$&CHR$(T)::NEXT I::A$=B$::B$=""
9040 SUBEND

```

This is a routine to get the date from the user. It forces the person to enter only the day, month and year to get: 01/01/87 with the slashes.

```

10000 SUB GETDATE(R,C,DATE$)
10010 IF DATE$="" THEN DATE$=" / / "
10020 DISPLAY AT(R,C)SIZE(8):DATE$
10030 FOR I=0 TO 2::ACCEPT AT(R,C+I*3)SIZE(-2):DT$(I)
10040 NEXT I
10050 DATE$=DT$(0)&"/"&DT$(1)&"/"&DT$(2)
10060 IF DATE$="//" THEN DATE$=""
10070 SUBEND

```

The next subroutine adds a certain number of spaces to the end of a string to make it a fixed length. This is to make reports and assignments line up. It works much in the same way as the print using but also works with assignment statements. Example:

```

>A$="HELLO":B$="THERE":A$=A$&B$::PRINT A$
HELLOTHERE
>A$="HELLO":B$="THERE":CALL PUTSP(A$,10)::CALL
PUTSP(B$,10)::A$=A$&B$::PRINT A$
HELLO·      THERE
<---10---><---10--->

```

here is the routine:

```
11000 SUB PUTSP(A$,SP)
11010 A$=A$&F-I$(" ",ABS(SP-LEN(A$)))
11020 SUBEND
```

Conversely, this next routine will remove any trailing spaces from the string. This is used for instance getting parts of a string:

```
>A$="HELLO     THERE":B$=SEG$(A$,1,10)::PRINT B$:LEN(B$)
HELLO
10
```

Notice that B\$ has the value of "HELLO" but it is 10 characters long. The DROPSP function will remove these trailing spaces:

```
12000 SUB DROPSP(A$)
12010 FOR I=LEN(A$) TO 1 STEP -1
12020 IF SEG$(A$,I,1)>" " THEN 12040
12030 NEXT I
12040 A$=SEG$(A$,I,1)
12050 SUBEND
```

LEARNING TI-99/4A ASSEMBLY - PART 2

By NICK IACOVELLI, JR

Well, part 2 is finally here. My articles are now being reprinted in the user group newsletter. Any questions to me should be mailed to:

ARTICLES
c/o NICK IACOVELLI, JR.
1411 N. 36TH
MELROSE PARK IL 60160

They may also be left for me on the TI-WEST BBS at 312-766-2797 for quicker response. All question will be printed and answered in the next article.

In the last article, I told you about the utilities and how to put a character on the screen. With this article, will show you how to display more than one character on the screen, how to move the character when a key is pressed, and how to fill the screen with that character (Call Hchar).

Last article showed this:

Last article showed this:

```
DEF START
REF VSWW
START LI R0,0
      INC R0
      BLWP @VSWW
```

This put a single character at screen location 400. Now, let us try filling up the screen with characters.

```
DEF START1
REF VSWW
START1 LI R0,0
       LI R1,>4100
       NEXT
       BLWP @VSWW
       INC R0
       CI R0,768
       JNE NEXT
       END
```

The LI R0,0 sets the screen location to 0.

The INC R0 adds 1 to the screen location because the next location I want to write to is 1.

The CI R0,768 checks R0 for maximum screen location. (Screen location is from 0 to 768).

The JNE NEXT will tell the machine to go to does not equal 768 and will write the next character on + screen. JNE means jump not equal.

WRITING A SENTENCE ON THE SCREEN

The new command is the TEXT command. This allows you to write a sentence and display it on the screen.

THE PROGRAM

```
DEF START2
REF VMBW
T1 TEXT THIS WILL SHOW UP ON THE
TEXT SCREEN
START2 LI R0,0 * SCREEN POSITION
       LI R1,T1 * LOCATION OF TEXT
       LI R2,34 * NUMBER OF CHARACTER
       BLWP @VMBW * DOES THE WRITE
       END
```

THE USE OF THE KEYBOARD

Just like the Call Key command in BASIC, Assembler works the same way with the BLWP @KSCAN Command. In order to map the keyboard in BASIC, a CALL KEY(#1,J,S) is used. The #1 determines which keys are active (more information is in your TI-BASIC book). In order to map the keyboard using assembler, we must move a 1 to a particular location in memory. This location is 8374. The way to move to that location is:

```
LI    R1,20100
MOVB R1,@8374
```

The LI puts the number we want in the memory. The MOVB (which stands for MOVE BYTE) move the first numbers to the location in the second half or loc 8374, in this case.

LET'S MOVE ONE CHARACTER EVERY TIME WE HIT A KEY

```
DEF  START3
REF  VSBW,KSCAN
START3 LI    R0,0          * SCREEN POSITION
      LI    R1,>4100      * CHAR 'A'
      BLWP @VSBW
HITKEY LI    R1,>0400     * MAP THE KEYBOARD
      MOVB R1,@>8374     * IN BASIC MODE
      BLWP @KSCAN
      LI    R1,2EE00
      CB   R1,2000,R1     * COMPARE R1 TO 2000
      JEQ  R1,2000,R1     * YES IT IS 2000
      LI    R1,>2000     * CHAR ' ' A SPACE
      BLWP @VSBW         * WRITE TO SCREEN
      INC  R0            * ADD 1 TO SCREEN POS
      CI   R0,768        * LAST POSITION
      JEQ  START3       * YES START OVER
      LI    R1,>4100     * NO WRITE TO NEW POS
      BLWP @VSBW
      JMP  -HITKEY       * GOTO HITKEY
      END
```

The 'C', 'CB', and 'CI' are compare commands. These commands are used with the 'JMP', 'JNE', 'JEQ', 'JGT', or 'JLT' commands. Think of these commands like the IF-THEN-ELSE statement in BASIC.

FOR EXAMPLE:

IN BASIC, A typical code would be;

```
10 REM
20 IF A=1 THEN 10 ELSE 30
```

IN ASSEMBLY, the code would be:

```

    A
    D1
    DATA >0001
    D1 DATA >0001

```

A and D1 would be labels to a data statement in assembly. They are put in the assembly like this.

```

    A DATA >0001
    D1 DATA >0001

```

You could change A or D1 by adding to it, by subtracting from it, or by moving to it. I will continue this in my next article and explain most of the instructions for moving data.

In response to STEPHAN MEYERS' article in the user group newsletter. NO FIRE STARTED BUT YOU MUST AGREE THAT FORTH DOES MAKE A RATHER MESSY PIZZA AND THAT MAKES YOU PERFECT TO DELIVER IT. AS FOR ME, ASSMEBLER IS A PIECE OF CAKE AND A LOT NEATER TO MAKE.

TI FORTH PART 2

Now that you have had some time to play with the system and maybe become comfortable with FORTH, It's time to take the options you like to use and turn them into a fast loading BSAVE program. We will add a couple of words that I use all the time, PAGE and FREE. PAGE is nice to clear the screen and sets the curser at upper left. FREE is a good one, it will tell you how much memory you have free. Maybe by now you have defined some of your own words.

```

    Lets define the word PAGE:
    : PAGE 0 0 GOTOXY CLS ;

```

Just as you see it on the above line. Thats all there is to it. You have a new word.

```

    Lets define the word FREE:
    : FREE SP HERE - .

```

Now you can see the amount of free memory anytime.

Now we have to find a place on a screen for these new words. You can pick a empty screen or add them to scr#3 later, because we have to update scr#3 anyway. The only option you need to put these words in the vocabulary is -SYNONYMS. So load -SYNONYMS and define these words and see

lets try work.

Lets do the job now. BSAVE. The first thing is to put the backup disk in drive 1. Better yet make another copy of the backup disk, because we are going to write on this disk and on some of the screens that you may want to look at in the future. Now we have a 3rd copy. The first is the original with the write protection tab on it. The second is the backup that has screen 72 corrected "the bug removed and set for PIO". Now lets work on the third. Put it in drive 1 and load FORTH. If you are already in FORTH, go to command mode and type: TEXT COLD press enter this will reboot the FORTH system to make sure we start from scratch. Now we have the flashing cursor under the word TI FORTH Lets load our options.

The ones you like to use are these, type:

-GRAPH -DUMP -VDPMODES -COPY -PRINT -BSAVE -64SUPPORT then press enter.

The -64SUPPORT or -EDITOR can be loaded depending on which one you like better. REMEMBER you can not use both.

In my system I have loaded everything but -EDITOR -TRACE -ASSEMBLER -CPU -FLOAT, to save memory. They can be loaded after the BSAVE is in memory.

After the ok is displayed and the cursor is flashing again we are ready to do our BSAVE type: TASK 51 BSAVE press enter (NOTE the first character in the line is the apostrophe) Now we have our fast loading program on disk. We just have to tell FORTH where it is at. type: EMPTY-BUFFERS 3 EDIT and press enter.

Now we have to change scr#3 to tell FORTH where the BSAVE is at and also put in a little more information. SCREEN#3 is the welcome scr. If you are using the 40 column editor you have line numbers, if -64SUPPORT you have no line numbers and have to count them for yourself. This is what scr#3 should look like after all changes.

```

0 ( WELCOME SCREEN) BASE->R HEX 10 SYSTEM
  ( Clears Screen)
1 0 0 GOTOXY ." TI FORTH BOOTING" CR 10
  83C2 C! ( QUIT off)
2 DECIMAL 51 BLOAD 16 SYSTEM MENU
3 1 VDPMODE ! ( Tells FORTH you're in TEXT
  mode)
4 0 DISK_LO ! ( Allows EDIT/COPY on all
  SCREENS)
5 89 DISK_HI ! ( Sets highest scr number)
6 89 DISK_SIZE ! ( Set single sided disk)
7 : FREE SP HERE - . ; ( Free memory)

```

```

8 : PAGE 0 0 GOTOXY CLS ; ( Clear screen)
9 : CLRF
10 :
11 :
12 :
13 :
14 :
15 R=PAGE

```

Now don't get the line numbers mixed up with the text, and don't type in the line numbers. The line numbers on in column 1, they are just there for reference. Now hold down the function key and press 9. Back to the command mode. type:

```

FLUSH
press enter

```

Our new screen is written and we can try it out. Type
 TEXT COLD
 press enter

That was quick, we have just loaded everything you saved with BSAVY

try it again, this time turn the computer off and start from the master screen. FAST ha?

Now all you need is the program "CLONE" and you can copy ANYTHING in TI language. You can do it also with the FORTH command "FORTH_COPY"

Most of this information I read from MILLERS GRAPHICS, "The smart programmer", 1475 W. Cypress ave. San Dimas, Ca 91773.



"HEY, MADE ANY PLANS FOR THE WEEKEND YET?"