99/4A OWNER/USERS GROUP

MONTHLY NEWSLETTER

Not affiliated with Texas Instruments, Inc.

APRIL 1984 VOLUME 2: ISSUE 4

DON'T FORGET

Our next meeting is April 27, 1984; at the Woodstock High School. This month voting on our bylaws will take place along with our regular program. Our program for this month will be... A Review of New Products to enhance our systems; A Look at some Tigercub Software; Home Computer Magazine giveaway and last but not least; The formation of some needed committees. We know that you will enjoy this meeting, so mark it on your calender and plan to attend. Meeting starts at 7:00 PM sharp.

EXPLORING FORTH with Bob Cwik

It has been two weeks since I received my copy of the FORTH system disk and made a copy of the manual. I have learned a few things since then and I would like to share them with you.

I got through the first two chapters of the TI FORTH manual without too much difficulty as these are self explanatory. Getting into chapter 2 which is the third chapter in the book (The manual starts with chapter 0) I immediately ran into trouble. There is an awful lot of good information in chapter 2 but 99% of it didn't mean much to me. They had a general overview of the FORTH language and syntax for experienced programmers so obviously this was not for me. There is, however, some very good advice in the begining of the chapter and that is to get a book "Starting FORTH" by Brodie.

In chapter 3 we got back into the real world or at least something I could handle, one step at a time instructions. Here we learned how to use the edit mode in FORTH and how to set up your own working disk. In my case I used a disk I had previously formatted with the Disk Manager. Then by following the directions faithfully, I was able to transfer Screens 4 and 5 from the System disk to my working disk. After that, things came to a grinding halt rather quickly. I could follow the use of the edit commands and even practice using them on screen 1 but I had no idea of what I was doing or where to go next.

At this point I realized I would need help so as soon as possible, like the next day, I went to explore the local book store and sure enough they had a book called "FORTH Fundamentals" by McCabe. It wasn't one of the ones referenced in the manual but it looked like it would be useful so I bought it and started to learn a new language.

In the first few chapters of the book I found myself very excited as they talked about the power and speed of FORTH and also very encouraged as they talked about stack operations and postfix notation (also known as RPN) as this is what I use on my HP calculator. For those of you not familiar with stack and postfix operations perhaps a brief explaination by example. In Algebraic notation (also known as infix) a problem such as (1+2)*(3-4) is worked out in a conventional manner. In FORTH the same problem would be entered like this: 1 2 + 3 4 - * . . The 1 and the 2 are first pushed into a stack followed by an operator, in this case the plus sign, which causes the top two items in the stack to be added and the result put back in the stack. At this point the only number in the stack is the result until the computer looks at the next item, identifies it as numeric and puts it on the stack on top of the last result likewise with the following item which is another numeric. Keep in mind there are now three numbers on the stack, the top number is a 4 which was last entered then a 3 which was next last entered and finally on the bottom is another 3 which was the result of the first operation. Now we come to the next operator which is a minus. It causes the top item to be subtracted from the next one down removing both from the stack and leaving only one item in the stack until it computes the result and puts that back on the stack. Now we encounter the next operator which tells us to take the top two items in the stack and multiply them together then put the result back on the stack. Now looking back at the original FORTH version of the

problem you will see two periods at the end of the sentence. No that is not a mistake. The first period is part of the FORTH language and it means to take the top number from the stack and display it on the screen. At this time there is only one number in the stack, that being our final result and after it is taken for display purposes it also disappears from the stack which would then be empty.

I can't even begin to explain everything I read about in the early chapters of FORTH Fundamentals. If you really want to learn about single precision, double precision, signed and unsigned numbers and stack manipulators such as DROP, DUP, SWAP, etc then your best bet is to go out and get the book. I have since gotten the recomended book by Brodie and it is definately clearer in it's presentation.

Let's end this first article with something useful. In the second item in the March Newsletter an error is described in SCREEN 72, here is how to make the correction:

1. From a cold start boot in your FORTH System Diskette.
2. Call up the Edit mode; enter -EDITOR
3. When OK apears enter 72 EDIT
4. When SCREEN 72 apears carefully move the cursor down to line 5 using FCTN X.
5. Carefully change PAB_ADDR to PAB-ADDR.
6. As long as you are in the neighborhood you might want to go up to line 4 directly above and change the device description to match your printer.
7. Exit the Editor with a FCTN 9 BACK.
8. The cursor should jump to the bottom of the screen under line 15.
9. Temporarily remove the Write Protect foil on your system diskette, replace the diskette in Drive one and enter FLUSH.

OK, now that you have had a good laugh, try it. And don't forget to replace the Write Protect foil. Next time, I hope to be writing short programs and maybe we can pick one of them apart in the Newsletter. See you at the meeting.

TIGERCUB SOFTWARE's Jim Peterson

The following is a response to an interview question we asked of Jim Peterson of TIGERCUB SOFTWARE. The question was..."Give me a little background on yourself and Tigercub Software." Here is his responce. We felt it both interesting and informative.

"There is not much to tell. Two years ago I had never seen a computer. My son, then a high school senior, was planning a career in engineering, so I thought it was time he learned something about computers. I bought a TI-99/4A, mainly because he liked it's keyboard the best. However, he was too busy with girls, sports, cars, girls, job, girls, etc., and never spent much time at the keyboard. I read the books that came with the computer, and tried to write a program of flags of the world. I soon became addicted. I never took any computer courses or had any training or advice from anyone else, but learned to program from reading the reference books and other books from the library, from translating the Microsoft Basic programs in Ahl's books of games, and by analyzing programs that I bought or swapped from the International User's Group. As a result, some of my programming techniques are self-invented, very unconventional, and difficult for other programmers to figure out - in fact, it is sometimes hard for me to figure out what I've done!

In less than a year, I had written some 90 programs. I am retired on disability, so I had plenty of time to spend at it. Then I helped establish a local User's Group, hoping to meet many other programmers and swap programs with them. I met very few programmers but plenty of promoters offering me all kinds of deals that seemed much more favorable to them than to me. So, I decided that the best way to get them to quit bothering me was to go into business for myself.

I picked the name Tigercub from TI and the "cub" meaning that I was offering small programs, not claiming them to be equal to the software being sold for $10 to $20. I picked a price of $3 because that was what the International User's Group was charging."

THE BASICS OF ASSEMBLY PROGRAMMING - PART 3

Last month we talked a little about number systems and how memory was addressed. We looked at the two's compliment form of representing numbers plus binary and hexadecimal notation. This month we shall look into the memory architecture of the 99/4A computer.

As you will see the memory is divided up into many segments which are reserved by the system for certain things. If taken all together the amount of total memory is 65536 bytes, or better known as 64K. Remember that the term "K" in base 16 (HEX) numbers is equal to 1024, and not 1000.

TI-99/4(A) MEMORY ARCHITECTURE

DESCRETE DEVICE	ADDRESS	USAGE
TMS9900 CPU	>0000	CONSOLE ROM (8K BYTE)
	>2000	MEMORY EXPANSION (8K BYTE)
	>4000	DEVICE SERVICE ROMS (8K BYTE)
	>6000	COMMAND MODULE ROM/RAM (8K BYTE)
	>8000	MAPPED PORTS (SEE BELOW)
	>A000	MEMORY EXPANSION (24K BYTE)
MAPPED PORTS		
FAST RAM	>8000	256 BYTES
TMS9919 SOUND	>8400	WRITE DATA
TMS9918A SCREEN	>8800	READ DATA
	>8802	READ STATUS
	>8C00	WRITE DATA
	>8C02	WRITE ADDRESS
TMS5200 SPEECH	>9000	READ DATA
	>9400	WRITE DATA
GROM CONTROL	>9800	GROM READ DATA
	>9802	GROM READ ADDRESS
	>9C00	GROM WRITE DATA
	>9C02	GROM WRITE ADDRESS

As you can see from the above memory table, the amount of usable memory in the BASIC's is limited to the 8K block at >2000 and the 24K block at >A000, for 32K total RAM. The rest of the memory (32K) is considered to be system overhead. This means that this is the needed amount of memory to carry on the required functions of the computer. The addresses given above are known as base addresses. These are the beginning of a block or segment of memory which contains a group of functions, such as the 8K block between >9800 and >9C02.

In assembly language programming, a method called base plus displacement addressing is used to calculate and notate internal addresses. Given a known base address, you need only figure the amount of offset, or displacement, needed to arrive at the desired address.

When counting bytes, start with zero as the address of the first byte. Zero is the first positive number to the

computer, so always begin counting at zero, not one. Consider a particular area of memory, VDP RAM. In VDP RAM, the first byte (byte zero) represents the first screen position. In TI BASIC, this would be row 1, column 1. One byte represents one character. For example, if the zero byte of VDP RAM contained the value >41 (decimal 65), the letter "A" would be displayed at row 1, column 1 of the screen. Hex >41 or decimal 65 is the ASCII code for the letter "A". It is by placing the correct values into this area of VDP RAM that symbols and graphics are made to appear on the screen. Most of the functions you will want the computer to do will involve placing certain values into specific areas of the computer.

Next month, the Screen Image Table, VDP RAM.

LETTERS TO THE NEWSLETTER

A reader asks - "What is the difference between GOSUB and the user definable CALL subroutines and how are they used?"

This is a very good question. One I'm sure many readers have wondered about from time to time, and one which most readers have had difficulty in understanding. We'll give it a try. TI EXTENDED BASIC offers in it's syntax the mainframe capability of called subprograms which have been written by the user into the current program in the workspace. These called subprograms share a remarkable similarity to the basic command called GOSUB nnnn, where nnnn is the line number of the beginning of a subroutine. Note a simple difference, subroutine and subprogram. It is the difference between these two terms that we seek to understand.

Simply stated, a subroutine is contained within a program and is a part of the main program. Subprograms on the other hand are not part of the main program, and are not executed by the RUN statement. Extended Basic programs which contain user written subprograms during RUN are scanned by the system and placed into memory just as other subprograms are. CALL CLEAR, CALL COLOR, and CALL CHAR are examples of these. They, however, are not executed until they are called. This process of initializing these subprograms occurs during the systems prescan of your program.

If there is any one thing unique about user defined subprograms it would have to be it's ability to pass parameters from the subprogram to the main program. This by itself does not seem to be such a great feat. However, add to this the fact that variables assigned within the subprogram are unique. For example, the variable "X" if used in a subprogram is not the same a "X" in the main program, unless it is the passed parameter.

Examine the example below...

```
100 CALL CLEAR
110 FOR N=1 TO 10
120 PRINT N
130 CALL EXAMPLE
140 CALL CLEAR
150 NEXT N
160 END
170 !
180 !
30000 SUB EXAMPLE
30010 FOR R=5 TO 15
30020 FOR C=5 TO 15
30030 CALL HCHAR(R,C,65)
30040 NEXT C
30050 NEXT R
30060 SUBEND
```

The program above will demonstrate the user defined subprogram "EXAMPLE". The next example illustrates passing parameters...

```
100 CALL CLEAR
110 INPUT "ENTER A NUMBER":X
120 IF X=0 THEN END
130 CALL CALCULATE(X,OUT)
140 PRINT "THE SQUARE ROOT OF ";X;"IS";OUT
150 FOR DELAY=1 TO 500 :: NEXT DELAY :: GOTO 100
160 !
170 !
20000 SUB CALCULATE(IN,OUT)
20010 OUT=SQR(IN)
20020 SUBEND
```

In this last example, the variable "X" was passed to the subprogram as the variable "IN" and returned as variable "OUT" to the main program. Remember, a subprogram is not a part of the main program even though it contains line numbers and is a part of your program listing. Given all this, then how does GOSUB nnnn differ from these user defined subprograms? Actually its quite simple. If a subprogram were written and then saved in the MERGE format, you could use this subprogram in many different programs. Heres the difference now, pay attention; A subprogram is called from the main program by name, not by line number.

If you don't catch on now, then perhaps you better stick with the old tried and true GOSUB nnnn.

A reader asks - "I spent quite alot of time writing a program that I wanted to save to a cassette tape. When I tried to save the program all I got was I/O ERROR 00. What Gives?"

Perhaps the key to your problem lies in the "spent quite alot of time". If your program was larger than 12K in size, then it could not be saved to CS1. Since there is no error massage built into the system to cover this condition, the computer just gives you the I/O ERROR 00 message.

Consider this, first printed in this newsletter (Vol.1: Issue 11: Page 2), and reprinted here from page 13 of the Extended Basic Addendum.

"The Memory Expansion unit adds 32K bytes of Random Access Memory (RAM) to the built-in memory of the computer. However, even with the Memory Expansion unit available, the largest TI Extended BASIC program that can be stored on a cassette tape is 12K bytes in size. Note that, although the length of the actual program is limited, utilizing the Memory Expansion unit provides other advantages. For example, with the unit attached and turned on, your program (which can be up to 12K bytes in length) is stored in the expansion RAM. The numeric data generated by the program is stored in the Memory Expansion unit and the string data is stored in the computer's built-in memory. Without the unit, the program must be shorter so that both it and the generated data can be stored in the computer's built-in memory."

CURRENT GROUP OFFICERS

Since next month is an election month we thought you might like to know who your user group officers were for the past fiscal year. Here they are at their incumbent best...

Bob Eckert - President, Newsletter Editor; (815)653-9341
Brett Pierce - Vice President; (312)669-3455
Laurie Britton - Secretary, Red Head; (815)344-4572
Editha Nelson - Treasurer; (815)338-5656
Tom Nigbor - Institutional Representative; (815)338-5331

The positions of Librarian and User Group Coordinator have been vacated.

EDITORS COMMENT

You know, I used to go to Kmart once a week or some other place that sold TI equipment just to see what was new that I might be able to add to my system. I don't go anymore. There is nothing new from TI. TI is out of the busniess. I miss them.

Lately, I've even shed a tear or two while writing this newsletter when I remember those days when products and software (really good software) was comming out almost every day. First rate stuff too! Its all gone now, and I feel alone. I care for my system now like it was made of gold, cause it cannot be replaced. I miss the link to them, the support I had, the pat on the back for a job well done. I miss them in my mail and on my phone, I still feel abandoned.... What's this! Somebody else is building for the TI computer. Another company is saving me, comming to my aid, lifting my spirits.

CorComp Incorporated to the rescue. This is it, the great pumpkin who watches over us. A smile, and then another smile. CorComp, CorComp?? It dosen't even sound like TI, what can they do for me? Can they put back the the lost spirit. Can they put back the anticipation we once had? Can they fill the shoes of TI? .....I wonder!

Susan? Yes Bob. Susan, can you tell me anything about a company call CorComp? In just a minute Bob. I'm working on the house temperature right now. I can't seem to stabilize it. Since you have seen fit to ignore my request to fix my No. 2 camera, I cannot see the front door to the house to see if your kids left it open. You know, Bob, those kids.....Susan, can we get back to my question? No Bob. We cannot get back to your question until I am able to secure the house, and this I cannot do at this moment. By the way Bob, you have been sitting on the stool in the bathroom for over 15 minutes, would you care to explain this? Upon observation, other members of the family use this facility with only an average of 3.6734 minutes duration. Are you in need of help? No Susan, I don't need any help, and its none of your business how long I spend in the bathroom, besides, it's something you would not understand. Now about my question! Close the front door! Very well Susan.

CorComp, nothing on file. Susan! Must you play with me when I need you? Susan?.....Susan?? COMMAND LEVEL 99, SUSAN DIAGNOSTIC RUN. Bob, I was only Kidd!.....SUSAN.....SYMBOLIC USER SYSTEM ANNUNCIATOR.....RAM CHECK.....OK.....ROM CHECK....OK.....I/O PORTS CHECK.....1...2...3...4...5...6..7.....I/O PORT 2 DISABLED.....SOUND CHECK.....OK.....SPEECH CHECK.....OK.....LOGIC CHECK.....nOK...END OF DIAGNOSTIC TEST.....ALL SYSTEMS OK.

Susan, about CorComp? Bob, I have nothing on file for this name. Susan, can you link to another computer to find out anything about CorComp? Bob, those big mainframes will rape me if I go snooping around. Susan! Your a computer. Bob, I want you to..... WHAT ARE YOU DOING! STOP, STOP, DON'T SHUT MY SYSTem dow.....

Rest Susan, Rest. Its going to be ok.

Just a little fix here and there. SUSan...How do you turn on the heat....manualy.....Damn.....

- Bob Eckert 653-9341