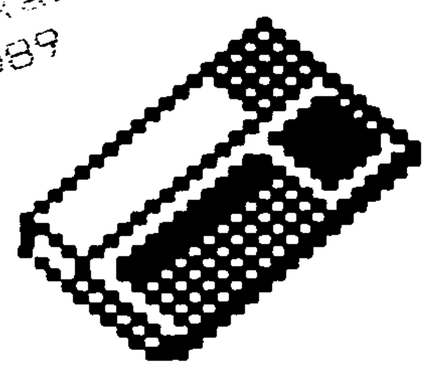
PROP. of HUG
SET "A"
c/o R.Lumpkin
Houston Texas
713-469-5089



HUG

HOUSTON USERS GROUP

SEPTEMBER
1985

MEETING SCHEDULE

FIRST SUNDAY OF EVERY MONTH
(2nd Sunday if 1st Sunday
is on a holliday weekend)

HUG TIBBS - (713) 699-2073
24-hour BULLETIN BOARD

****

AT THE NEXT MEETING

SUNDAY, SEPTEMBER 8, 1985 2:00 P.M.

St. John's School - 2401 Clairemont

The September HUG meeting program : Making use of the TI-99/4A in the business world. The program will demonstrate how the software can be used that is presently available. (see page 2)

****

IN THIS ISSUE

****

By the time you read this report, I hope you have received your copy of our proposed constitution and "Official Ballot" mailed to you a few days ago. If for some reason you did not receive it, please advise me. My appreciation goes out to the Constitution Committee and Sub-committees for all the work they put in on this proposal. I remind everyone to vote and turn in your ballot by 2:00 p. m. Sunday, September 8. The reason for this cut-off time is so the votes may be counted during the program and the results announced at the meeting.

Our picnic last month was a success, even though we would have like to had more of you participate in it. My thanks to Mark Crump for the effort put in on this project. Through Mark we learned that we needed a permit to hold this picnic in a city park, and he arranged for this, plus reserved us a covered area in Hermann Park. We were honored to have Past President Marshall Gordon of the Atlanta Users' group attend our picnic. He and I were able to discuss our groups and exchange a few ideas. It is nice to have visitors from other groups, so if any of you happen to be going out of town, check to see if there is a meeting in the town you are visiting and go out and visit with some other users.

Hopefully, our "summer slump" is now about over and we will be filling all the chairs again. We still have some excellent programs to bring to you this year and need your support. Rumors are still circulating about a compatible computer coming out. I'll believe them and report on the computer after I type my first "RUN" on it.

Hope to see all of you at the meeting.

Bill W. Knecht

This month we will have an old friend join us. Wayne Wright. As most of you old timers of HUG know, he is our former president who was responsible in getting this club going at a time when TI was getting out of the home computer business. His topic will be on the use of the TI99/4A computer in the business world.

Many of us bought our computers to use at home for games, education and to set up our household budgets. (Anyway that was the intent at the time.) But now how many have placed the computer in a corner and look at it only once in a blue moon. I know I fall in that category also. Well this little "Orphan" can be made useful in a lot of ways and Wayne has profitted from it as well. He runs his own business utilizing a number of TI99/4A's, and he will tell us a little of what it is like to use a computer that everyone said was only a toy in running a business. Based on my knowledge and association with Wayne, his remarks should be interesting as well as informative.

On another topic: Coming up with a program to present to the members. If you all have noticed the people who have presented programs in the past, you will have noted that they are the ones who were elected to office and the ones who take care of the business of running the club for the benefit of its "silent" members. You may have also noticed that it is these same people all of the time that come forth with the energy and the willingness to give of themselves. We have many takers and very few givers. What is my point? Im looking for input as to what type of program the general membership would like to see at the monthly meetings. So far the response that I have been given has been very sparse, at best. Either I am fantastic at guessing what everyone's interests are, or there is no interest. Let me know what you use your computer for. Is it just word-prossessing and playing games, or are there other things that this expensive'sil ver box is good for? If I have the input, then I can try to come up with an appropriate output (program). Let me know.....SANDUK

September HUG meeting program: Making use of the TI99/4A in the business world.

```
SCR #98
 0 ( 6 PASS SINGLE-DRIVE DISK COPIER 1/27/84)
 1 ( enter COLD, 98 LOAD, and DUPLACATE )
 2 BASE->R DECIMAL -SYNONYMS 0 VARIABLE BIG 15358 ALLOT
 3 : ?# EMPTY-BUFFERS 0 BLOCK 10 + @ 256 1024 #/MOD SWAP 0= 0= + ;
 4 : PAK CR ." PRESS ANY KEY" 52 GPLLNK KEY DROP CR CR ;
 5 : LMD ." LOAD MASTER DISK" PAK ; : LCD ." LOAD COPY DISK" PAK ;
 6 : DUPLICATE CLS 0 0 GOTOXY LMD 0 DISK_LO ! ?# DUP DUP
 7 DISK_SIZE ! DISK_HI ! LCD
 8
 9 0 DO CLS 0 0 GOTOXY LMD
10 I 15 0 DO DUP I + DUP . CR BLOCK BIG I 1024 # + 1024 CMOVE LOOP
11 CLS 0 0 GOTOXY LCD
12 15 0 DO DUP I + DUP . CR BLOCK BIG I 1024 # + SWAP 1024 CMOVE
13 UPDATE FLUSH LOOP
14 DROP 15 +LOOP 1 DISK_LO ! ; R->BASE
15
```

FORTH TO YOU. TOO!
SESSION 2
by
Lutz Winkler

IN THE LAST LESSON:

You have determined which of the editors suits you and found a display color you like. They could be entered from the key board each time FORTH is booted. But there is a better method: Let the disk do it for you! To begin with we'll use the simple - and later on a more elegant - way. (If you haven't made up an overlay yet, better do it now, else editing isn't going to be easy. Programming in FORTH is done by editing SCREENS and the various editing functions are made a lot easier if you can refer to the overlay.)

So boot your FORTH disk again and when the MENU shows up, enter either -EDITOR or -64SUPPORT. Now get out your manual and go to Appendix I More (Q)uit Hit ENTER to continue) (Contents of the Disk) and look at SCREEN 3. This is the one that gives you the first inkling that something is going on by displaying "BOOTING". So you get an idea of the way FORTH works, let's scan its content before going on :

Line 0: The parenthesis ( ) act like a REM in Basic, so we see that it is called the Welcome Screen. GOTOXY is like DISPLAY AT, note the coordinates 0 0 preceding it.

Line 1: Forget the BASE->R for now, but let's do something with HEX. From your keyboard enter :

HEX 83C2 DECIMAL .

Don't forget the period, actually a FORTH WORD called DOT. (Look up each word in the GLOSSARY) What did you get ? -31806 is correct. In plain English line 1 states: Switch to BASE 16, put >10 (16) on the stack, and C! (C-STORE, see page 17, Glossary) it at More (Q)uit Hit ENTER to continue) 83C2. This is how FORTH does the CALL LOAD for FUNCT-Quit Ctl. (You have seen that one before!)

Line 2: DECIMAL returns us to Base 10, ignore the (94 LOAD), 20 LOAD loads SCREEN 20 (look at scr # 20 and you'll see that it's the menu which appears at boot time. 16 SYSTEM is CALL CLEAR (more about System Calls later) and finally MENU displays the menu. Take a moment to digest this, as it gives some idea as to how FORTH works. The command 20 LOAD booted scr # 20 at which time a new FORTH WORD was compiled (see scr 20, line 1). MENU is now part of the DICTIONARY. Anytime MENU is invoked, FORTH looks it up and executes it. Try it, enter MENU. You get the menu and 'ok'. If you enter something FORTH can't find you'll see a '?', sometimes followed by an error message (see Appendix H). Most mistakes made by beginners are More (Q)uit Hit ENTER to continue) simple ones, such as missing spaces, colons or semicolons or a LOAD OPTION not booted.

OK, back to the Welcome Screen. But now let's put it on display. Enter 3 EDIT and watch it come up. Skip to line 4 and note that here we have the menu words defined, i.e. : -EDITOR 34 LOAD ; etc. The first word after a ':' is the new word being added to the dictionary. Any words that follow will be executed, provided FORTH can find them in the dictionary. The definition ends at ';'. Now move the cursor down to line 12 and type -EDITOR (or -64 SUPPORT) (enter). Are you surprised that nothing happened (except the cursor moved to the start of the next line)? That's because you are in the EDIT mode. If you are sticking with the normal Editor type in the number which you selected with the SEE experiment as you display color followed by ? WAIT. If you More (Q)uit Hit ENTER to continue) chose the 64-column Editor don't' bother, type : COLD TEXT COLD ; instead.

Hit the ESCAPE key (ESC) to get out of EDIT. Your additions to scr 3 are NOT actually on the SCREEN but in a buffer and you must enter FLUSH before going on.

Remember that every time you EDIT a SCREEN you must FLUSH, otherwise all your efforts will be for nought.

So let's check if your edit was successful. Enter COLD. This word is like NEW except you don't have to do anything else. FORTH will re-boot. (It'll take longer now because you are booting the editor also.)

Now let's recap:

You have 'edited' SCREEN 3 so it boots your editor and sets up the screen color for you. This was done while in the EDIT mode. You have also More (Q)uit Hit ENTER to continue) worked in the 'interactive' mode when you defined the word SEE to determine your color choice. In this mode you can try out your definitions before you use them in a program. You'll find this to be tremendously helpful because unlike BASIC there is no need to go to RUN and see what happens and then finding the line which needs to be changed.

Having worked my way into TI-FORTH the hard way, I will leave you with a few suggestions which I feel will be helpful:

Look up each new word in Appendix D of the manual. See how it is defined.

Mark the chapters and appendices in your TI-FORTH manual for easier access to them. You'll be using it frequently because - even though it may not seem so at first - it DOES contain a lot of information. to them. You'll be using it frequently because - even though it More (Q)uit Hit ENTER to continue) may not seem so at first - it DOES contain a lot of information.

Get a FORTH book, preferrably Leo Brodie's STARTING FORTH. It is sold in many bookstores/software houses. The manual (Appendix C) explains the differences between fig-FORTH, which Brodie uses, and TI's implementation of it.

Though it may read like greek, scan through the manual. As we go along you might just remember having seen something that rings a bell. (Finding it again may be something else!)

If you have any problems, feel free to call me at (619) 277-4437. I am usually at home evenings after 5 PST.

End Session 2

```
SCR #110
 0 ( SIMPLE SKETCH PAD PROGRAM MODELED AFTER VIDEOGRAPHS 1/6 )
 1 ( DEFINE CONSTANTS )  BASE->R DECIMAL
 2  32 CONSTANT Y-MIN
 3 191 CONSTANT Y-MAX      ( THESE ARE MIN AND MAX FOR )
 4   0 CONSTANT X-MIN      ( FOR SPLIT2 MODE )
 5 255 CONSTANT X-MAX
 6   0 CONSTANT FALSE      ( BOOLEAN TRUE / FALSE )
 7  -1 CONSTANT TRUE
 8
 9  ( DEFINE VARIABLES )
10  2 VARIABLE MODE        ( CURRENT DRAWING MODE )
11  0 VARIABLE LASTKEY     ( ASCII VALUE OF LAST KEY PRESSED )
12  0 VARIABLE SAMEKEY     ( FLAG USE TO SIGNIFY SAME KEY HELD )
13  0 VARIABLE DELAY       ( DELAY SIZE )
14  R->BASE -->
15
```

```
SCR #111
 0  ( SIMPLE SKETCH PAD PROGRAM CONTINUED )
 1  ( DEFINE UTILITIES USED ) BASE->R DECIMAL
 2  : NOT  0= :                      ( DEFINE 'NOT' FUNCTION )
 3  : 2DROP DROP DROP :       ( 2DROP REMOVES 2 NUMBERS FROM STACK )
 4  : TEST4SAME  ( TEST TO SEE IF SAME KEY PRESSED KEY # ASSUMED )
 5           ( TO BE ON TOP OF STACK AND IS LEFT THERE UNCHANGED )
 6            DUP LASTKEY @ =       ( IS NEWKEY SAME AS LAST KEY )
 7            IF TRUE SAMEKEY !     ( YES: SET SAMEKEY TO TRUE )
 8            ELSE DUP LASTKEY !    ( NO : SET LASTKEY =NEWKEY )
 9            FALSE SAMEKEY !       ( AND SET SAMEKEY TO FALSE )
10            ENDIF :
11  : TEST_DRAW  ( TEST TO SEE WHICH DRAWMODE WE ARE IN )
12             ( IF MODE IS NOT = 2 DRAW A PINT AT POINTER )
13             MODE @ 2 = NOT IF 1 SPRGET DOT ENDIF ;
14   R->BASE -->
15
```

```
SCR #112
 0  ( SIMPLE SKETCH PAD )
 1  ( DEFINED SUBROUTINE USED ) BASE->R HEX
 2  : DEFINE_CURSOR ( DEFINE POINTER USED TO DRAW WITH )
 3             3800 ' SATR ! 3800 SSDT ( INITIALIZE SPRITES )
 4             0F0E0 0E090 0804 0201 0A SPCHAR ( DEFINE POINTER )
 5             20 20 0F 0A 1 SPRITE :  ( NOW DESPLAY SPRITE )
 6  DECIMAL
 7  : DO_MODE ( ROUTINE TO CHANGE MODES ( 0=DRAW 1=ERASE 2=MOVE )
 8           MODE @ CASE 0 OF 2 MODE !        ." MOVE "   ENDOF
 9                       1 OF 0 MODE ! DRAW   ." DRAW "   ENDOF
10                       2 OF 1 MODE ! UNDRAW ." ERASE"   ENDOF
11                       ( OTHERWISE ) 2 MODE !   ENDCASE ;
12  R->BASE -->
13
14
15
```

```
SCR #113
 0  ( SIMPLE SKETCH PAD )
 1  ( DEFINE SUBROUTINE USED CONT ) BASE->R DECIMAL
 2  : DO_MOVE ( JOYST-X JOYST-Y --- NEW-X-POS NEW-Y-POS )
 3            ( TAKE JOYSTICK READING READ IN READ_JOYST AN
 4            ( CONVERT THEM INTO A MOVEMENT OF THE POINTER  )
 5            1 SPRGET  ( GET THE CURRENT SPRITE POSITION )
 6           ROT 4 / - Y-MIN MAX  Y-MAX MIN   ( DO Y POSITION )
 7           SWAP ROT 4 / + X-MIN MAX X-MAX MIN ( DO X POSITION
 8           SWAP ; ( RESTORE PROPER ORDER FOR X AND Y )
 9   : TEST-4-FC  ( N  --- N )
10                ( SINCE OUTPUT FROM JOYST IS A BYTE # WE MUST
11              ( TEST FOR THE NUMBER 252 AND REPLACE IT WITH --
12              ( IF NUMBER = >FC REPLACE IT WITH >FFFC  )
13              DUP 252 = IF DROP -4 ENDIF :
14              R->BASE -->
15
```

```
SCR #114
 0  ( SIMPLE SKETCH PAD PROGRAM )
 1  ( DEFINE SUBROUTINE USED CONTINUED )  BASE->R DECIMAL
 2  : READ_JOYST  ( READ JOYST, DECIDE TO CHANGE CURSOR OR MODE
 3                1 JOYST ROT  ( READ JOYST AND PUT KEY# ON TOP
 4                TEST4SAME CASE 18 OF SAMEKEY @ NOT IF DO_MODE
 5                ENDIF 2DROP ENDOF 255 OF TEST-4-FC SWAP
 6                TEST-4-FC SWAP DO_MOVE 1 SPRPUT ENDOF ENDCASE
 7
 8  : HOME 24 0 GOTOXY ;
 9  : DO_DELAY  DELAY @ 0 DO LOOP ;
10
11  R->BASE -->
12
13
14
15
```

```
SCR #115
 0  ( SIMPLE SKETCH PAD PROGRAM )
 1  ( MAIN ROUTINES )                BASE->R
 2  : DO_KEYBD ?KEY CASE 0 OF    0 DROP           ENDOF
 3                  ( 1 )  49 OF    0 DELAY !     ENDOF ( SELECT
 4                  ( 2 )  50 OF  100 DELAY !     ENDOF ( DELAY
 5                  ( 3 )  51 OF  200 DELAY !     ENDOF ( TIME
 6                  ( 4 )  52 OF  400 DELAY !     ENDOF
 7       ( SPACEBAR )    32 OF SPLIT2 DEFINE_CURSOR ENDOF ( CLEAR
 8                  ENDCASE ;
 9  HEX
10  : SKETCH-PAD SPLIT2 DEFINE_CURSOR  ( INITIALIZE )
11         1 0 DO ( BEGIN MAIN LOOP )
12             READ_JOYST DO_DELAY TEST_DRAW DO_KEYBD HOME
13             ?KEY 02 = IF LEAVE ENDIF ( TEST FOR EXIT CONDITION
14         0 +LOOP ( END OF MAIN LOOP )
15         DELALL HEX 300 ' SATR ! DECIMAL TEXT :     R->BASE
```

```
SCR #116
 0  THIS IS A SIMPLE SKETCH PAD PROGRAM FOR YOU TO TYPE IN AND
 1  ENJOY. THE SOURCE OF THE PROGRAM IS A HAND OU GIVEN TO ME BY
 2  BOB LAWSON. THE AUTHOR IS JEFF STANFORD. JEFF MOVED TO FLORIDA
 3  JUST RECENTLY. ( SOUND FAMILIAR DOESN'T IT ! ) I HAVE NOT BEEN
 4  ABLE TO REACH HIM IN REGARDS TO THE PROGRAM. I AM SURE THAT
 5  HE WILL APPRECIATE ANY COMMENTS AND ATTA-BOYS ON HIS WORK.
 6
 7  EDITORS NOTE : THE PROGRAM LISTING THAT YOU SEE HERE WAS
 8  PRINTED DIRECTLY FROM FORTH SCREENS. THIS IS DUE TO THE WAY
 9  TI-WRITER HANDLES SOME OF THE SYNTAX USED IN FORTH. I KNOW
10  THAT MANY OF THE ARTICLES AND PROGRAM LISTINGS FOR FORTH ARE
11  GOOFED UP BY OTHER NEWSLETTERS BECAUSE THEY AUTHORS CHOOSE TO
12  USE TI-WRITER TO PRINT THE PROGRAMS OUT. THIS ONLY CREATS
13  PROBLEMS. IF YOU RUN THE PROGRAM AND FIND ANY DIFFICULTIES
14  DOUBLE CHECK YOUR SCREENS. THE LISTINGS ARE FROM A PROGRAM
15  THAT WAS WORKING OK AND WAS A DIRECT PRINT-OUT OF THE SAME.
```

```
SCR #117
 0  TO RUN THE PROGRAM. YOU MUST DO  THE FOLLOWING.
 1
 2  LOAD      -GRAPH  -SPLIT  -TEXT
 3            AND THE BEGINNING SCREEN NUMBER OF YOUR PROGRAM
 4  THEN TYPE     SKETCH-PAD  AND PRESS ENTER
 5
 6  BE SURE THAT YOU HAVE USED CONSECUTIVE SCREENS WHEN YOU
 7  TYPE IN THE PROGRAM. ALSO. BE SURE TO TO TYPE ALL OF THE
 8  SCREENS THAT YOU SEE HERE  ( EXCEPT # 116 ) SO THAT ALL OF
 9  THE INFORMATION THAT YOU NEED IS SOMEWHERE ON THE DISK.
10
11  JOYSTSTICK # 1 IS USED PRESSING THE FIRE BUTTON CHANGES MODES
12  ( DRAW, MOVE AND ERASE ) PRESSING 1, 2, 3, OR 4 SELECTS
13  DRAWING SPEEDS WITH 1 BEING THE FASTEST. THE SPACEBAR CLEARS
14  THE SCREEN. FUNCTION 4 EXITS THE PROGRAM.
15
```

TUTORIAL 1 - THE BEGINNING
by Mack McCormick
Downloaded from Compuserve
Consumer Information Service

Here are the objectives of this first tutorial: 1. To introduce you to the Hexadecimal and Binary Numbering systems. 2. To introduce you to the assembler instruction format. 3. To introduce you to addressing modes. 4. First program. Adding two numbers and displaying them on the screen. 5. How to assemble.

Just a few words on Assembly language before we begin. It's not as difficult as you may believe. You will be communicating with the microprocessor at the first level above machine language, assembler. As you know, the machine actually communicates in binary 0's and 1's, on or off. Assembler allows us to talk to the machine in a language we can understand (Although I'm sure the uninformed would disagree). With these tutorials I will assume no prior knowledge of assembler or other number systems. Please bear with me. I won't (More (Q)uit Hit ENTER to continue) insult your intelligence and things will become more complex soon. Stick with the tutorials. Read every book about assembler you can get your hands on. I will publish a bibliography of books soon. Don't get discouraged! Compuserve is a difficult medium thru which to provide assistance. I promise to answer your questions and if I don't know the answer, I'll find someone else that can. Please make this an interactive process, as we grow and learn with each other.

Numbering Systems

Hexadecimal (HEX) and binary are merely different base numbering systems for counting. It's important we understand both of these systems in addition to base 10 or the decimal system because assembler uses all three. I will tell you up front that I use a calculator designed for these numbering systems usually but we need to understand the principles also. If you want to get a calculator, and I recommend that you do, there are several inexpensive models on the market. I use the Casio solar powered fx-451 scientific calculator for $35. It supports HEX, OCT, BIN, LOGICAL OPERATORS, and all scientific functions. Works great! Craig Miller and others have also published programs which will allow you to use your computer but this has the disadvantage of requiring you to load another program every time you need to make a calculation, a real pain.

Binary Number System

As already mentioned, binary is the native language for your computer. Everything eventually gets converted to binary. Let's look at a decimal number first. As you know decimal means powers of 10. Each number represents a power of ten. For example 4175:

10 3=1000 4 X 1000=4000

10 2= 100 1 X 100= 100

10 1= 10 7 X 10= 70

10 0= 1 5 X 1= 5

4175

Binary numbers can be 1 or 0 only, hence base 2. The individual number is called a bit. A group of eight of these is called a byte. To convert the binary number 00001011 to decimal follow the same procedures you used with the decimal number:

Ignore any leading zeros.

2 3=8 1 X 8=8

2 2=4 0 X 4=0

2 1=2 1 X 2=2

2 0=1 1 X 1=1

11

To make it easier to communicate with the computer we most often use HEX. From now on I will use a > to indicate a number is in hex. Hex is base 16. That is a number may be 0 thru 15. To represent numbers greater than 9 we use letters of the alphabet. 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F. Just remember to use >A for 10 and count to 15 ending with >F. Let's convert >394F to decimal:

16 3=4096 3 X 4096=12,288

16 2= 256 9 X 256= 2,304

16 1= 16 4 X 16= 64

16 0= 1 F X 1= 15

14,671

The largest number you may represent in one byte is >FF or decimal 255. The largest value in a word (two bytes) is >FFFF or 65,535. Enough on numbering systems for now, we'll cover minus numbers (twos compliment) and additional points as we encounter them in programs.

Assembler Instruction Syntax.

Like every computer language there are certain rules we must follow for inputting instructions. Unlike BASIC, assembler will not give you a warning or error until you assemble the program. Here's the general syntax:

LABEL OPCODE OPERAND COMMENT

Labels must begin in the 1st column and may be up to 6 characters long. One or more spaces follow. Next is the OPCODE. This is the actual instruction to be performed followed by one or more spaces. Next are one or more operands or data for the instruction to operate on followed by one or more spaces. Finally is an optional comment which may extend to column 80. Each time we use a new instruction I will fully explain it.

Addressing Modes.

There are five general addressing modes and one special addressing mode used for assembler instructions. We will examine each one in detail as we encounter them in a program. There is one type of addressing we need to look at now. We are going to be operating on individual bits, bytes, and words of memory. Think of the computers memory as a series of consecutive small pieces of memory laid out end to end. We can address any single byte of memory by hanging a label on it but frequently we must address a byte of memory some distance from that label. Think of it like an array. To get to the 5th byte from the label we could say LABEL+4. We used 4 instead of 5 because we must start counting from 0. Think of it like OPTION BASE 0 in BASIC. Let's more on this later.

First Program. I strongly recommend you enter the program manually by typing it in instead of just cleaning it up using TI-Writer or Editor/Assembler. The only way to gain experience programming is to practice.

I've place the program separately to make it easier to read. It's is in Pt2 in the (I)nfo section.

Program explanation . These comments supplement the comments contained in the program itself. Any statement with an * in column 1 is a comment and you may More (W)uit Hit ENTER to continue) enter anything else on that line. One fairly unique thing about the 9900 microprocessor in the TI-99 is the ability to designate your own workspace registers anyplace in memory or more than one set at a time. Think of registers as 32 consecutive bytes of memory that are used as your scratch paper for calculations. Thirty-two bytes is of course 16 words of memory. Because this is a 16 bit (1 word) machine (something many of your friends can't brag about) that gives us 16 registers to use for our computations. We place an R in front of the number to designate that we are refering to a register. For example, R0 is register zero and R15 is register fifteen (really the 16th word of memory because we started counting with zero.) Here's the detailed explanation of the program:

DEF START

DEFines the entry point of the program so the computer may find it. Places the name START in the Reference/Definition table. More on this next time.

REF VSBW,VMBW

REFerence refers to console routine the program will use. In the advanced tutorials we'll create our own utilities.

```
*****************************************
*                                       *
* THIS IS THE FIRST PROGRAM FOR THE     *
* BEGINNER ASSEMBLER TUTORIAL.          *
* IT CLEARS THE SCREEN, ADDS TWO NUMBERS *
* TOGETHER AND DISPLAYS THE SUM IN THE  *
* CENTER OF THE SCREEN.                 *
* HERE'S WHAT IT WOULD LOOK LIKE IN     *
* EXTENDED BASIC:                       *
* 10 CALL CLEAR                         *
* 20 X=10                               *
* 30 Y=27                               *
* 40 X=X+Y                              *
* 50 DISPLAY AT(12,15):X                *
* 60 END                                *
*****************************************
*
* THIS PART OF THE PROGRAM IS THE INITIALIZATION *
*
       DEF  START  THE PROGRAM NAME IS START
       REF  VSBW,VMBW CONSOLE ROUTINES WE ARE GOING TO
*                     USE
WSREG  BSS  >20     SETS ASIDE A BLOCK OF 32 BYTES FOR USE
*                   AS WORK REGISTERS
X      DATA 10      CAN USE >A INSTEAD OF 10 (LIKE X=10)
Y      DATA 27      (Y=27) COULD HAVE ALSO SAID Y EQU 0027
TEN    DATA 10
ANS    DATA 0       WORD TO PUT ANSWER IN. INIT TO 0.
*
* PROGRAM BEGINS HERE
*-----------------------------*
START  LWPI WSREG   LOAD WORKSPACE POINTER IMMEDIATE
*                   TO POINT TO OUR WORK SPACE
* CLEAR THE SCREEN
       CLR  R0      CLEARS R0 TO ZERO (BEGINNING OF
*                   SCREEN IMAGE TABLE )
       LI   R1,>2000 LOAD IMMEDIATE R1 WITH >2000. VSBW
*                   ROUTINE WRITES THE LEFT BYTE IN R1
LOOP   BLWP @VSBW   ALWAYS. THIS TIME >20 OR 32 OR SPACE
```

```
SCR #108
 0            HERE IS A GLOSSERY OF TERMS FOR FORTH
 1 FUNCTION 1 .....................DELETE CHARACTER
 2 FUNCTION 2 .....................INSERT CHARACTER
 3 FUNCTION 3 .....................ERASE LINE
 4 FUNCTION 4 .....................MOVES TO NEXT SCREEN
 5 FUNCTION 5 .....................SHOWS OTHER HALF OF SCREEN
 6 FUNCTION 6 .....................GO BACK ONE SCREEN
 7 FUNCTION 7 .....................NOT USED
 8 FUNCTION 8 .....................INSERT LINE
 9 FUNCTION 9 .....................DROPS CURSOR TO BOTTOM OF
10                                   SCREEN FOR COMMAND ENTRY
11                  COMMANDS
12 EMPTY-BUFFERS  ..................CLEARS THE MEMORY BUFFERS
13 FLUSH          ..................SAVES EDITED SCREEN TO DISK
14 (SCR #) EDIT   ..................MOVES TO THAT SCREEN FOR
15                                   EDITING OR ENTRY OF PROGRAM

SCR #109
 0                       COMMANDS CONTINUED
 1 (SCR # ) CLEAR     ..... CLEARS SCREEN OF ALL DATA
 2 (SCR # ) SWCH LIST UNSWCH  .. WITH THE PRINT SCREENS
 3                               LOADED. PRINTS THE SCREEN
 4                               TO THE DEVICE SPECIFIED ON
 5                               SCREEN 72
 6 (SCR # ) LOAD   ..............LOADS THAT SCREEN INTO MEMORY
 7 ED@     .....................RETURNS TO THE EDIT SCREEN
 8                               YOU WERE ON BEFORE ENTERIN
 9                               THE COMMAND MODE
10 FORGET xxxxx  ................DELETES THE SPECIFIED DEFINITION
11                               xxxxxx AND ALL DEFINITIONS AFTER
12                               IT THAT HAVE BEEN ENTERED
13 FORMAT-DISK   ................FORMATS A DISK FOR USE WITH A
14                               FORTH COPYER
15
```

```
        INC  R0         ADD 1 TO R0
        CI   R0,>67     COMPARE IMMEDIATE R0 TO >67
        JLE  LOOP       IF ITS LESS THAN OR EQUAL JMP (GOTO)
*                       LOOP
* ADD THE NUMBERS TOGETHER AND CONVERT TO ANCII
*-------------------------*
*
        A    @Y,@X      ADDS X TO Y AND PLACES RESULT IN X
        MOV  @X,R6      MOVE WHATS AT X TO R6
        CLR  R5         CLEAR R5
        DIV  @TEN,R5    DIVIDES 10 INTO 37. QUOTIENT IN R5.
*                       REMAINDER R6.
        AI   R6,>30     ADD IMMEDIATE ANCII OFFSET TO R6
        MOV  R6,@ANS    MOVE CONTENTS OF R6 TO THE WORD ANS
        MOV  R5,R6      MOVE CONTENTS OF R5 TO R6
        CLR  R5         CLEAR R5
        DIV  @TEN,R5    DIVIDE 10 INTO R5. R6
        AI   R6,>30     ADD IMMEDIATE ANCII OFFSET >30 TO R6
        SLA  R6,8       SHIFT LEFT ARITHMETIC R6 8 BITS.
        MOVB R6,@ANS    MOVE MSBYTE R6 TO R1
*-------------------------*
* DISPLAY ON THE SCREEN AT ROW 12 COLUMN 15
*
        LI   R0,366     POSITION ON THE SCREEN IS 366
        LI   R1,ANS     LOAD R1 WITH THE ADDRESS OF ANS
        LI   R2,2       TWO BYTES TO WRITE
        BLWP @VMBW
        JMP  $          PREVENTS THE PROGRAM FORM ENDING SO
*                       YOU MAY SEE THE RESULT
*-------------------------*
* RETURN TO THE CALLING PROGRAM
*
        CLR  @>837C     CLEAR THE STATUS BYTE
        LWPI >83E0      LOAD GPL WORKSPACE REGISTERS
        B    @>0070     BRANCH TO THE CALLING PROGRAM
        END
```

```
SCR #91
 0 ( 3 PASS DISK COPIER DOUG SMITH 301-645-1432)          : IT ;
 1 : CLS 16 SYSTEM ;  : VMBW 2 SYSTEM ;  : VMBR 6 SYSTEM ;
 2  0 VARIABLE AREA 15360 ALLOT 0 VARIABLE PL 0 DISK_LD !
 3 : TX CLS 5 11 GOTOXY ;
 4 : M1 TX ." INSERT COPY DISK-PRESS ANY KEY " KEY DROP ;
 5 : M2 TX ." INSERT  MASTER - PRESS ANY KEY " KEY DROP ;
 6 : M3 TX ." DONE - ENTER N TO COPY ANOTHER " ;
 7 : PR TX ." COPIER NOW READY-PRESS ANY KEY " KEY DROP ;
 8 : PU PL @ 20 + PL @ 5 + DO I BLOCK AREA 2 + I PL @ 5 + - 1024
 9      * + 1024 CMOVE LOOP ;
10 : BU PL @ 5 + PL @ DO I BLOCK UPDATE LOOP M1 FLUSH ;
11 : DR PL @ 10 + PL @ 5 + DO AREA 2 + I PL @ 5 + - 1024 * + I
12      BUFFER 1024 CMOVE UPDATE LOOP FLUSH PL @ 15 + PL @ 10 + DO
13      AREA 2 + I PL @ 5 + - 1024 * + I BUFFER 1024 CMOVE UPDATE
14      LOOP FLUSH PL @ 20 + PL @ 15 + DO AREA 2 + I PL @ 5 + -
15      1024 * + I BUFFER 1024 CMOVE UPDATE LOOP FLUSH ;     -->
```

```
SCR #92
 0 ( 3 PASS COPIER CONT )
 1 : BPU PL @ 28 + PL @ 20 + DO I BLOCK 5120 I PL @ - 20 - 1024
 2       * + 1024 VMBW  LOOP
 3       PL @ 28 + BLOCK 3072 1024 VMBW
 4       PL @ 29 + BLOCK 1122 1024 VMBW ;
 5 : BDR PL @ 25 + PL @ 20 + DO 5120 I PL @ - 20 - 1024 * + I
 6       BUFFER 1024 VMBR  UPDATE LOOP FLUSH
 7       PL @ 28 + PL @ 25 + DO 5120 I PL @ - 20 - 1024 * + I
 8       BUFFER 1024 VMBR  UPDATE LOOP
 9       3072 PL @ 28 + BUFFER 1024 VMBR UPDATE
10       1122 PL @ 29 + BUFFER 1024 VMBR UPDATE FLUSH ;
11 : PAS BPU PU BU BDR DR ;
12 : CY  0 PL ! 30 / 1+ 0 DO PAS PL @ 50 > IF M3 ELSE M2 I 2+
13       30 PL +! THEN LOOP ;
14 : W   M2 89 CY ;
15 PR W
```

HUG LIBRARY CATALOG ADDENDUM
July-August 1985

3045 TI PILOT##P-CODE REQUIRED
Texas Instrument's teaching language. May be used to program your own Plato programs. REQUIRES 1 SSSD DISK. Documentation will be made available for you to copy or you will be charged the current price per copy. Documentation is approximately 70 pages front and back. 358 sectors.

4110 TI KEYBOARD LETTERHEAD TI-Writer & Printer Rqd.
Print your own name and address on letterhead with TI Keyboard Logo at top of page. 7 sectors

4111 ITOD W/DOCUMENTATION##XB Printer optional.
Makes Multiplan Sylink files accessible from Basic or Extended Basic. E/A or TI-Writer needed for documentation. An excellent program by Mark Crump. 14 sectors.

4112 DM1000-W/DOCUMENTATION##XB
Disk Manager program on a disk distributed by Freeware and programmed by Bruce Caron. This program has many excellent features such as disk protection and unprotection, initializing a whole box of disks , and the ability to recover a lost file. DEDICATED DISK RECOMMENDED. Documentation can be printed out on TI-Writer or EA Module. 227 sectors

4113 HBMPRINT##FORTH##EA MODULE REQUIRED. Printer required
A Freeware program by Bob Lawson which figures home budget management. Program is written in Forth and extremely fast. Documentation on disk under file name READ-ME. Use TI-Writer or EA Module to print out. REQUIRES DEDICATED DISK 322 sectors

4114 XMODEM-W/DOCUMENTATION##EA Rqd.
Terminal emulator program written by Paul Charlton. Can be used on such BBS systems as COMPUSERVE. 169 sectors

4115 TE3MN-W/DOCUMENTATION##EA Rqd.
Revised version of Program #4056. Revised 6/9/85. Load & Run TE3. 149 sectors

4116 FAST-TERM-W/DOCUMENTATION##EA Rqd.
Terminal emulator program written by Paul Charlton and distributed as "Freeware". Excellent program. DEDICATED DISK RECOMMENDED 242 sectors

4117 DB/CONVERT##XB
Program by Scott Medbury to convert sound to decibel ratings. 5 sectors

5215 SECRETLY##XB
Music & nice title screen of a Letterman hit by Bill Knecht. 20 sectors

5216 PLEASE PLEASE ME##XB
Music & colorful title screen of this Beatles' hit of the 60's. One of Bill Knecht's best. 21 sectors

5217 NEW/KID##XB
Eagle's hit of "New Kid in Town" redone by Bill Knecht. Excellent music and graphics. 51 sectors

5218 RISING-SUN##XB
Good version of popular classic "The House Of The Rising Sun" by The Animals. Excellent program with graphics. 20 sectors

5219 CANTINA##XB
Cute graphics and music from Star War's Cantina scene by Edwin McFall. 49 sectors

M & S COMPUTER SYSTEMS
15918 CAVENDISH DRIVE
HOUSTON, TEXAS 77059

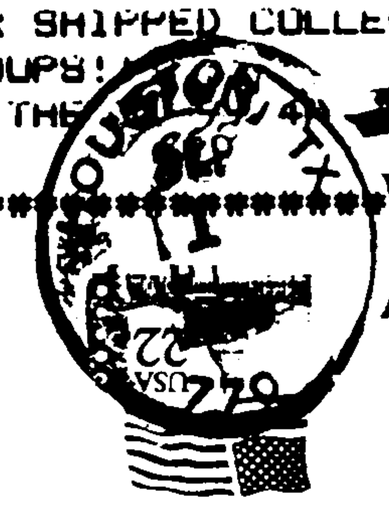
SEPTEMBER SPECIALS

STAR SG-10 NEAR LETTER QUALITY DOT MATRIX PRINTER, 120 CPS.....	$ 249.95
VOLKSMODEM 12, 300/1200 BAUD DIRECT CONNECT MODEM W/ CABLE.....	225.00
CORCOMP TRIPLE TECH CARD W/ CLOCK/CALENDER, 64K PRINTER BUFFER..... AND SPEECH SYNTHESIZER CONNECTION	114.95
CORCOMP 9900 STAND ALONE CLOCK/CALENDAR W/ LOAD/INTERRUPT SWITCH....	73.95
DATABIOTICS "4A/TALK" TERMINAL EMULATOR, 110-9600 BAUD, W/ X-MODEM.. & TE II PROTOCOL. CAN DOWN LOAD/UPLOAD TO OR FROM DISK. DIALER PROGRAM INCLUDED!, X/B, E/A OR M/M, 32K, DISK DRIVE	17.95
DATABIOTICS "BITMAC", #1 RATED GRAPHICS PROGRAM, X/B, E/A, OR M/M... 32K & JOYSTICK OR TRACKBALL, DISK DRIVE	31.95
MILLER GRAPHICS "ADVANCED DIAGNOSTICS", SUPER UTILITY PROGRAM,..... X/B OR E/A, DISK DRIVE, 32K	17.75
MILLER GRAPHICS "EXPLORER", AN EXTREMELY EASY UTILITY INSTRUMENT TO. USE WITH ITS FUNCTION AND CONTROL KEYS, BINARY SWITCHES, AND ITS FULL SCREEN EDIT CONTROL FIELDS. INCLUDES FUNCTION KEY STRIP, EXTENSIVE 100 PG MANUAL PLUS MORE!	21.95
99-WRITER II(TEX COMP)HAS ALL THE FEATURES OF TI WRITER, X/B, E/A... OR M/M, 32K, DISK DRIVE	19.95
BANNER-99(TEX COMP), CREATE YOUR OWN BANNERS, SIGNS & BUMPER..... STICKERS, REQUIRES X/B, 32K, DISK DRIVE & PRINTER	19.95
BONUS DISKETTES, LIFETIME WARRANTY, SSDD, 11/ BOX 2 BXS(22 DISKS)	19.95
FLIPPY FLOPPY DISKETTES, DSDD, 10 PACK, W/O PLASTIC LIBRARY CASE....	12.95
DISK BANK MEDIA MATE 5, HOLDS 50 DISKS.....	11.75

WE SHIP UPS, TEX-PACK OR US MAIL. ORDERS CAN BE PREPAID OR SHIPPED COLLECT

WE WELCOME PHONE & MAIL ORDERS FROM OTHER TI USER GROUPS!

WE CARRY A LARGE VARITY OF TI & THIRD PARTY PRODUCTS FOR THE 99/4A



Sanford & Ruth Herman
6219 Sanford Road
Houston Tx 77096