



Atlanta
99/4A
Computer
Users
Group

CALL NEWSLETTER

VOLUME IV NUMBER 7

SEPTEMBER 1986

Atlanta, Georgia

PRESIDENT'S CORNER

Those that ordered the diskettes by the hundred (50 if you went halves with someone) can pick them up at the October meeting.

No one took Boyd Cone of Information Associates up on the offer to order merchandise ahead of the Swap Meet at distributor prices. I was a little surprised but it is understandable. I feel that the best benefit that could have come from it would be for those that order equipment and not software. A few did consider it that I know of, but held off. Others asked me about how to order, but the items they were considering were in the under \$10 range and was software that has been vastly reduced already. I suggested to them that those goods were likely to be found at the Swap Meet anyway.

So... come to the November Swap Meet and bring another. Tell other folks about it.

Remember we are preparing to have MORE hardware and software available at this one than there was at the one in July. If you have things to sell then bring them. Half the idea is for people to sell what they don't want or have outgrown, and the other half is for those coming to find bargains, of which there should be plenty. Over a dozen people at the last meeting said they would help put up notices which are being prepared.

Speaking of looking to buy. Hal Kam needs to find the modulator that hooks between the computer and the TV. It seems Delta Electronics no longer has them at all; much less for \$2 like they were for a time. Anybody know of a supply? In the same vein. I have always wondered if those TV/Game converters sold in department stores could be made to work.

Changing the subject around to the newsletter:

The bulk of this newsletter was typed in by Bobby Miller who is willing to co-ordinate assignin files for people to type and then send him by modem. Jim Barksdale has typed some and a bit back Melvin Carter did some heavy typing. We did not take proper advantage of what Melvin typed and it may have discouraged him.

The procedure now for getting files from people who type and putting them into the newsletter seems to be working. The last several newsletters show it. If you are willing to type contact Bobby Miller.

That brings up an obvious point. Our newsletter has of late been mostly articles from the other groups. Bobby suggested that it was not at all bad for our newsletter to pick and choose so we could present some of the best for our members and others to see. To make the most of the DOZENS (a little less than a hundred) of these newsletters we need people to read, digest, and type in for our printing. If this is done the size of ours will grow back to where it used to be at 16-20 pages.

We could put one out now that easily has 30+ pages IF we just -cut and paste- directly from those that we receive. That however would be TACKY as well as just plain WRONG. It is sad that with few exceptions we had not had much original material to publish this year; but if we are to continue as we have done of late then lets do it up right and make it look as good as we can. Anyway back to the present. This issue has some articles that go together well and we hope they will be of use to you.

A last note about Home Computer Magazine or Journal or whatever since by now even the most hopeful are having to admit it has died a slow and painful death. One of our members, Melvin Carter, has written the Postal authorities to complain about their practices. I have heard that the Post Office has had numerous complaints about HCM. If you call the HCM offices you will be told that the organization has shut down and whoever you manage to speak to really is not associated with that past organization; even though they just happen to be the same people. Apparently a get tough act works for those that are desperate or just plain angry enough to feel they must demand their money back. At this point most are calling their money lost. It wasn't a lot of money and a number of people I have spoken to are not really angry, just disappointed that a few more issues weren't published.

Gary Matthews

***** CLUB OFFICERS *****

 Gary Matthews President
 404-233-3096
 Jim Hubbard Vice President
 (1)345-5905 w-482-9421
 George Sears Treasurer
 396-4112
 Melvin Carter Secretary
 997-2617
 Marshal Gordon Newsletter Chairman
 Bobby Miller Newsletter Coordinator
 921-8308
 Ed Banovatz Library Chairman
 872-4088
 Jim Fairchild BBS(991-6250) System Operator
 Charles Dupree BBS(366-1914) System Operator

CALL NEWSLETTER

CALL NEWSLETTER is the voice of the Atlanta 99/4A Computer Users Group, P.O. Box 190841, Atlanta, GA 30325.

It is published at least 10 times a year. The A9CUG is not affiliated with any commercial company or organization.

CALL NEWSLETTER is published by and for the members of the A9CUG to enhance their knowledge of home computers. It is composed of articles written and/or donated by members of our group and from articles appearing in other home computer users groups around the world. Opinions expressed by the authors do not necessarily represent those of the officers or other members of the A9CUG.

Permission is hereby granted to any users group to reproduce any article appearing in this newsletter, unless the article is otherwise noted, provided credit is given to the author and the originating user's group. The A9CUG freely exchanges newsletters with other groups around the country. If another group would like to receive our newsletter but does not have one of their own to exchange; we will gladly send it to them. We do ask that they send \$5 a year to cover costs.

Membership is open to family and individuals who own or are interested in using and programming home computers. Membership includes newsletters as they are printed, access to meetings, and membership privileges. Annual dues are \$15.00.

ALTERNATE DRIVE INTERFACES
by Tom Burke of Phila BBS

If you are thinking of adding a second drive to your system, and it is not a TI drive, it may or may not work by connecting it to the external drive connector on the back of the Disk Controller Card. If it does Not work with the TI recommended installation procedure, try this.... You will need the following: an adequate length of 34 conductor ribbon cable. (2 to 3 feet) 1-female 34 pin connector for ribbon cable. This will connect to the controller card inside the P>E> box. Part #41-908 6c

It should look something like this. 2 - female 34 contact edgcard connectors. part #41-946 This makes a total of three connectors. The part #'s are from RESCO Electronics Use the crimp-on type connectors. Put part #41-908 6C on one end of the cable. Plug this into the controller card inside the pe box. Place the first edge card connector approx. eight to twelve inches from the controller card. Crimp it on the cable. Now you should loosen the screw on the right, rear, top of the outside of the PE box enough so that you can slide the other end of the cable out through the gap. When this is done, plug the middle connector onto the internal drive, (don't forget to plug the power cable back on the drive), and put the drive back into the PE box, pulling the excess cable through the gap. DO NOT INSTALL THE DRIVE PERMANENTLY AT THIS TIME. JUST LEAVE IT SIT IN THE PE BOX. Next, crimp the remaining connector on to the other end of the cable. Bare in mind that when you do put the connectors on the cable, that the same wires must be connected to the same numbered contacts on all three connectors. (e.g. wire one goes to pin one on connectors 1,2,3) Plug the last connector onto the external drive. (contact one to contact one, etc) Now test the system. You may or may not have to remove the resistor from drive #1. Info on where the resistor is found in the controller manual.

WHAT IS A NIBBLE, ANYWAY?

BY JIM SWEDLOW

FROM THE USERS GROUP OF ORANGE COUNTY

This month I am going to try and explain all of the various number words we run across. With luck, after you finish reading this, you will have some understanding of bit, byte, nibble, word, hex, binary, and where -31952 really is in memory.

With luck.

Computers really think in binary. In this numbering system there are two numbers, 0 and 1 (or, if you are a computer, off and on). While this works for your 4A, binary is cumbersome for humans. For example, in binary 41,576 is 101000110001 1100.

Hex, or hexadecimal, has sixteen numbers from zero to F. Here are the first sixteen numbers in binary, decimal and hex:

DECIMAL	HEX	BINARY
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

The next number would be 16 or >10 or b10000 (> means hex and b means binary).

One binary number is a bit. Four bits is a nibble. So, 10, A, or 1010 takes four bits or a nibble to express.

A byte is eight bits or two nibbles. With a bit you can count from zero to one. A nibble gets you from zero to fifteen. The range of byte is:

BASE	LOW	HIGH
Binary	0	11111111
Hex	0	FF
Decimal	0	255

You have probably noticed the numbers 16 and 255 when using your TI. ASCII character run from 0 to 255. There are sixteen colors (1 to 16, really 0 to 15). A string can be up to 255 characters long. And on and on.

Before tackling the next thing, a word, lets see if we can decode something. Lets take b10100 or >14. To convert either number to decimal, we need a method:

>14 is >10 plus >4
 >10 is 16 and >4 is 4
 16 plus 4 is 20
 Hence, >14 is 20

b10100 is b10000 plus b100
 b10000 is 16 and b100 is 4
 16 plus 4 is 20
 b10100 is 20

Futher than that I cannot go in this space. A word is sixteen bits or four nibbles or two bytes. The range of a word is:

BASE	LOW	HIGH
Binary	0	1111111111111111
Hex	0	FFFF
Decimal	0	65,535

But there are no negative numbers. Since we need them, we use something called twos compliment ! (which is way beyond the scope of this column and this writer). I can tell you, however, the impact;

Hex range	Decimal Range
0-7FFF	0 to 32,767
8000-FFFF	-32,768 to -1

Remember that >8000 is the next number after >7FFF.

Some examples:

7FFF is 32,767
 8000 is -32,768
 FFFF is -1
 0 is 0

Confused? So was I until I worked with it for a while. These conversion rules may help:

>>Any number less than or equal to 32,767 requires no conversion.

>>Subtract 65536 form any number over 32,767.

>>Add 65536 to any number less than zero.

This conversion process can be expressed in basic:

AD=AD+65536*(AD>32767)

If AD is the address, this returns the same number if AD is less than or equal to 32767. If AD is greater than 32767, the test returns true (-1) and a negative 65536 is added to AD. Try it on your computer.

Bottom line time. Suppose you see CALL PEEK(-31952,A,B). Where is -31952? Well, since it is less than zero, we add 65536 and get 3584 or >8330. NOW YOU KNOW!

MACHINE CODE MASTERY

Articles from the people at FUNNELWEB FARM

Editor/Assembler
~~~~~

The ultimate way to get at the real potential of the TI-99/4a is to write or run machine code programs. The next best thing is TI-Forth, but that's grist for another mill. Originally TI did not intend that users would ever write their own machine language programs and provided no hooks at all in console Basic to link to machine language routines, or to allow direct access to machine functions. The same sort of corporate marketing contempt for the customer led them to put calculator keys on the original TI-99/4. They weren't and aren't alone in that attitude of contempt for the user .... look at the IBM PCjr years later, or the Apple Macintosh.

And when they did bring out the expansion system, it still did not realize the potential of the TMS-9900 processor because of the fractured memory map and conversion of the 16 bit data path to successive 8 bit slices for all but a small part of CPU memory space, adding that insult to use of external memory with wait states. However when TI finally made machine code available to users they did it in style, adapting their mini-computer software for the purpose. Some of the programs supplied still contain traces of their origin, such as memory mapper instructions relevant only to the larger 990 minis.

First let's look at how machine code programs are recorded as disk or cassette files, and then survey the modules which allow these files to be loaded and executed. These files come in two forms. The most direct form is as memory image files, in which the actual contents of a block or blocks of memory are stored, with control information appended. These are known as PROGRAM files in TI-99/4a language (and correspond to .COM or .CMD files in other systems). TI Basic and XB programs are also stored in this format, which can be saved to and loaded from cassette as well as disk. The other kind, usable only with a disk system, is the assembly tagged OBJECT file. In normal usage of the TI-99/4a, object files are created to be relocatable in memory by the loader, and the programmer does not have to know explicitly where the loader has placed them, and calls their entry points up by name; None of the primitive USR or suchlike business. There are 4 (maybe 5) modules available which can load and run machine language programs. They all have different capabilities and limitations.

The primary one is EDITOR/ASSEMBLER which is necessary for creation of relocatable object files (... well you could write one with a word processor but that would be masochism of a high order, exceeding even direct POKEing of machine code bytes). E/A will load any of the machine code file types mentioned above, from its menu screen. The LOAD AND RUN option handles both uncompressed and compressed tagged object files, and will resolve REFERENCES by name from one object file to another, or to standard system names. Uncompressed object files represent bytes in Hex notation, and take about twice as much disk space as compressed object files. Invocation of LOAD AND RUN re-initializes the memory pointers completely while loading the system utility routines such as VMEW from GROM storage, so if a sequence of file loads is interrupted by an error, it must be started all over again.

E/A adds CALL LOAD and CALL LINK to console Basic to allow these same object files to be loaded and accessed from Basic. The standard utilities such as NUMREF for communicating with the Basic program must be loaded as a separate file BSCSUP.

E/A will also load and run from the RUN PROGRAM FILE option, program files of machine code, prepared according to a specific recipe as SAVE files. The details of these will be a subject for HV99 News articles in the future. It will load them from cassette as well but I can't see anyone doing that in preference to using disks, unless perhaps they have installed the TIUP internal memory expansion mod in the spare console that gets taken away on holidays. No provision exists in Basic to load SAVED program files from Basic as they could overwrite part of the Basic program in the VDP on the way in.

Extended Basic  
~~~~~

The next module which can load assembly code is our old friend EXTENDED BASIC. This is much more limited than E/A in what it will handle. Firstly it recognises only the 8K low memory area from >2000 to >4000 for loading relocatable object code. Absolute code can be loaded, or RAM buffer areas used in the lower part of high memory once it is known how far down this is filled by the XBasic program. The loader does not handle external REFERENCES, and the utilities

loaded by CALL INIT in XB are missing the most useful one - DSRLNK, and GPLLNK as well. The Basic support utilities are loaded by CALL INIT from GROM. The assembly source code has to locate them with EQUates. A minor difference from E/A is that CALL LINK always hands over control in the GPL workspace. Programs written to LINK to E/A Basic will almost always need at least minor modifications to LINK to XB successfully. The operational hangup with XB is that the loader is written in GPL and is painfully slow. A long assembly routine, such as Text to Speech, may take several minutes to load (shades of the Commodore 64's disk system). The usual way round this is to load an assembly language loader which in turn does a faster load of the longer program. The great virtue of the XB module that sets it apart from the others is that it supports auto-RUNning from disk, as soon as the module is selected, of an XB program DSK1.LOAD which can then load further programs. The other reason for preferring XB is that it is a vastly more powerful language than the mildly enhanced console Basic offered by E/A. Unlike E/A it can never load machine code programs without Basic as an intermediary.

Mini-Memory

This module has its own particular charm as the only one which allows access to machine code without the 32K memory expansion and using cassette storage. In this mode a LINE by LINE (or immediate input) Assembler allows standard TMS-9900 mnemonic assembly code to be entered in a restricted format. This is a descendant of TI's board level 990 evaluation systems. Only 700 odd bytes are available in the module's 4K of CMOS RAM after loading the assembler but I can't imagine anyone wanting to do programs longer than that with L by L. Still it's enough to do a pretty fair Game of Life program. MM also contains a full set of system utilities and Basic support routines in ROM and EASYBUG in GROM, a monitor program that is useful but much less powerful than the E/A DEBUG.

MM is even more useful in a fully expanded system. It does not provide the Editor and Assembler features of E/A, but offers more scope for loading and running programs.

Firstly there is 5-6K more RAM available, 4K of CMOS RAM in the cartridge and the saving of space in RAM because of the utilities in cartridge ROM. Its principal deficiency as compared to E/A is the lack of a PROGRAM file loader, but this can be easily remedied by writing your own to reside in MM cartridge RAM. Even the L by L Assembler, as well as EASYBUG, remains useful for occasional little purposes anywhere in RAM, and I have prepared a disk based version for convenience.

E/A object code, even compressed, is loaded successfully as long as REFS are used for system utilities and Basic support routines. EQUates as used for XB code will only get it right for one module. The loader has one more space, cartridge RAM, to place relocateable object code as a last resort. I have not yet experimented to see whether the loader will link object files with external references as the E/A loader does. The MM manual, never a fount of information, is silent on this point.

MM does not erase its DEF table unless it is explicitly done by one of several means. The table survives a return to the title screen, and even switch-off if the internal battery is still alive. This is different from E/A's workings, and must be taken into account for better or for worse. Code in memory expansion does not survive switch-off even if its name lives on.

TI-WRITER

Now we leave the modules which can load files under any name for one which loads program files with particular names. TI-WRITER tries to load an E/A type SAVE file from DSK1 under the name EDITAL (and successor filenames) when the EDITOR option is selected from the menu screen. If you have followed the E/A manual's advice on using the SAVE utility with TOMBSTONE CITY as the victim, take the file so generated and place it on a fresh disk under the name EDITAL and place in DSK1. Then fire up TI-WRITER, choose the Editor option, and see what happens. Extension to Formatter and Utility options are obvious. It may provide light relief to heavy TI-WRITER sessions. More seriously, short of writing a PROGRAM file loader to be loaded by XB using DSK1.LOAD, it is the nearest that the TI-99/4a comes to an auto-loader for machine code program files.

TI DISK OPERATING SYSTEM
Courtesy TI SOURCE (tm) BBS

Terminology

TI DOS denotes standard TI Disk Operating Systems.

Nibble or nibble-4 bits

Byte or bite-8 bits

Word 16-bits

RPM/S-Revolutions per minute/sec

FDC-Floppy disk controller

TRN-Termination resistor pack

SI-Sector Index (hole)

RW-Read Write

WPN-Write Protect Notch

OC-Optical Coupler

1S (or SS)-Single sided

1D (or SD)-Single density

2S (or DS)-Double sided

SI-Sector Index (hole)

RW-Read Write

WPN-Write Protect Notch

OC-Optical Coupler

1S (or SS)-Single sided

1D (or SD)-Single density

2S (or DS)-Double sided

2D (or DD)-Double density

General

A disk drive is a complex and precise piece of hardware, both mechanically and electronically. The disk drive spins at 300 rpm, positions a RW head, and communicates with the FDC. Disk accesses are in a semi-random fashion; although tracks are accessed at random, the Sectors are accessed sequentially, as the disk spins.

General TI

TI-DOS supports both 1S1D or 2S2D formatting. The latter, only if you have other than TI drives; (generally Shugart). You may have noticed, that, on the DMII, the options are available to format up to 2S2D. This is because TI was in the process of producing a 2S2D controller prior to its demise.

However, Corcomp has come through, and produced an excellent FDC. Using this, along with a couple of half-height TEACS, gives the capability of 2S2D formatting, A total of (approx) 369K per disk, of online storage is possible. How did I arrive at this figure? Well, let's break it down.

Normal TI drives (1S1D)

40 tracks 9 sectors per track and 256 BYTES per sector=90K bytes(approx). 2S1D would double that figure; 180K 2S2D would double again, 180K*2=360K per disk. Of course, not all of this disk space is available for programs and data, but neither is the 1S1D TI format. This will become clear in later tutorials.

TRN and Shunt Paks

The purpose of the TRN is to "pull up" the inactive FDC lines to +5v at the drive inputs. For optimum performance, these TRNs should be removed from all but the last drive. Some drive cables have gaps designed into their connectors to "open" unused drive select lines. The "shunts", however must not be removed, as these select, among other things, the drive #. One hint here. For convenience sake the shunt paks may be replaced by 8-position MINI-DIP switches. This will make re-configuring the system easy. TEAC drives, however, have relocatable jumper plug which makes things even easier.

Floppy Disks General

A mylar disk is contained within a protective sleeve. It is coated with mixtures of ferrite oxide(magnetic), lubricants, and binders. This disk spins freely inside the envelope. The envelope itself is multipurpose. It:

- a) Protects
- b) Supports(some degree of rigidity)
- c) Cleans the disk (as it spins, dust particles are forced into the lining).

The envelope has five cutouts. These are:

- 1) Alignment notches. Located on the bottom of the diskette . The diskette is put into the drive notches first. This seats the disk in the drive such that it is properly aligned with the internal components of the drive.

- 2) RW Notch. Located in the upper or left side of the disk(viewed while the disk is positioned for insertion), and is sometimes called the Write-Protect notch or WPN. With the disk in the drive, an optical detector picks up light which originates on the opposite side of the drive, and allows the disk to be written to. If this notch is covered, the disk cannot be written on.

3) Hub hole. This is in the center of the disk and allows the drive to "grab" and spin the diskette.

4) SI hole. Located near the center and punched on both sides of the envelope. As the disk spins, an optical detector picks up light from the opposite side of the disk, and, when the small hole (on the disk itself) is aligned with the SI holes, the FDC is informed that the disk is positioned at sector 00 (of whatever track is being accessed). Hence it serves as a survey marker for all other sectors on that track.

5) Long oblong window, cut into both sides of the envelope, is where the RW head presses against the magnetic material from one side, and a felt pressure-pad on the other (for single sided drives).

The RW head is positioned by a precise stepping motor which places the head to access tracks. Tracks are numbered (generally) from outside to inside. When a disk is "initialized" or formatted, these tracks are defined, as are the sectors.

Hard sector soft sector

Most computers use soft-sectored disks, including TI. A hard sectored disk is one which has one hole marking each sector of a given track (i.e. one hole designates the start of sector 01 of tracks 1-9). It also has a SI hole, which, again, marks sector 00. To determine if a disk is hard-sectored, merely turn the disk within the sleeve, and, look at the SI hole. As you turn the disk, there should be only 1 hole for soft sectored disks, and many holes for hard sectored disks. If the latter is the case, you will notice 3 holes are in close proximity to each other. This is the sector 00 marker. What's the advantage of hard sectored disks? Not much, other than more data can be stored on them due to the sector headers not having to store sector information. There are other minor advantages, but not worthwhile to consider.

FDC-Floppy Disk Controller

Disk drives are controlled by the FDC chip, which is a microprocessor in its own right. It is responsible for transfers of data to/from the CPU and disk drives, communicates its requirements to the RW head, and maintains status information (sector/track, etc) in its internal registers. The actual sequence of operation of the FDC is far too complex to detail here, and would be general at any rate, and

USING YOUR NEW DISK

By Darren Leonard from PUG

It has been drawn to my attention that a considerable number of our members are encountering some difficulty getting the programs on the disks of the month to run. In this column I will attempt to provide you with a concise and foolproof method to get any program running so long as it is not written in forth or pascal.

The first and most logical thing to do is catalog the disk using a disk manager. Your catalog should look like the following:

```

Diskname: PUGDOM9/85
Used 340 Free 20
Program      Size      Format P
-----
MYFILE       22      PROGRAM
MYFILE2      69      INT/VAR 254 (EXT BASIC)
MYFILE3      49      DIS/FIX 80
STARGAZER    33      PROGRAM (E/A #5)
STARGAZERS   33      PROGRAM (E/A #5)
STARGAZET    9       PROGRAM
SPACEGAME    47      PROGRAM
DATAFILE     87      DIS/VAR 80 (TI-WRITER)
SCRABBLE#1   57      INT/FIX 123
GAME#1       60      DIS/FIX 80 (E/A #3)
GAME#2       46      DIS/FIX 80 (E/A #3)

```

These are most of the possible file types you might receive.

1) Any DIS/VAR 80 type file like "DATAFILE" above are usually text or documentation files that should be read with an editor such as TI/WRITER or the Editor/Assembler package. Alternately, you can use this short Basic program:

```

10 OPEN #1: "DSK1.DATAFILE"
20 OPEN #2: "PIO" (or your printer's
commands)
30 IF EOF(1) THEN 70
40 INPUT #1:X$
50 PRINT #2:X$
60 GOTO 30
70 CLOSE #1
80 CLOSE #2
90 END

```

2) Any file with a size greater than 50 and saved INT/VAR 254 is a long Extended Basic program that requires memory expansion. To use this program, you would OLD DSK1.MYFILE2 RUN from XBasic.

3) Any file in the format DIS/FIX 80 is an Assembly language program that can be loaded several ways. The surefire way is to

use the LOAD and RUN option of an E/A cartridge or a MiniMemory Cartridge. You might need the Program name to start it. If it is not given to you, search the last 5 sections of the program with DISKO to look for it. Usual program names include: START, BEGIN, BOARD, GAME, LOAD, RUN, ect. If you do not have an E/A or MMM then you MAY be able to load it with Extended Basic. Memory Expansion is a MUST. You also need the program name to do this. Here is an example of an Extended basic program to run MYFILES3.

10 CALL INIT

--initializes memory expansion and prepares for loading

20 CALL LOAD ("DSK1.MYFILES3")

--Loads the program into memory

30 CALL LINK ("START")

--Starts the program IF start is the program name.

This will work most of the time, but not always. Therefore, your best bet is to use an E/A or MMM if you have one.

4) Any file that is in program format and is EXACTLY 33 sectors in size is most likely an assembly program file that will load out of option #5 of the Editor/Assembler module or the Utility option of TI/Writer.

5) Files that are program format and are longer than 33 sectors are either Basic or Extended Basic. Usually try Extended Basic first. If you get an error, try Basic. It might be necessary to free extra memory in console basic if you get a memory full error. To do this type:

CALL FILES (1)

NEW

OLD DSK1.SPACEGAME

RUN

If you still get memory full error, then it should be an extended basic program, or will only run on tape without the disk drive attached.

6) Any file in a program format that is less than 33 sectors could be a Basic, Extended Basic, or Assembly program. Try them in that order.

7) Any file that is in Program format and is 52 sectors long could be a Tunnels of Doom file. If you have the Tunnels of Doom Module, try it. If it is 54 sectors long, it could be an adventure game: if you have the Adventure module, try it.

8) When you have 2 consecutively named DIS/FIX 80 files, you must load the second file BEFORE you give the program name.

9) Any other type of format is most likely a data file that is used by a program on the disk. They will not run, unless it is a merge format, but in general should be left on the disk with the other programs.

A LOOK AT PROGRAMS

Article by R.A. Green from the Ottawa 99/4 User Group via the DELAWARE VALLEY USERS GROUP.

There are seven different ways to store programs in the TI 99/4A. In this article we will have a look at each of these seven forms and at how they are used.

Everyone is familiar with the form used by TI BASIC to store programs on cassette or disk. It's identified as "PROGRAM" in the disk catalog. It is created or stored by the BASIC "SAVE" command and loaded by the BASIC "OLD" command. This is the only way that TI BASIC uses to store your programs.

EXTENDEND BASIC can, and usually does, use the same form as TI BASIC to store programs. There are, however two other forms that XB uses. Both of these forms can only be used to store programs on disk.

If you have the 32K Memory Expansion, you can write a XB program which is too large to store in the usual format. XB will store these large programs in an "INTERNAL VARIABLE 254" file. The usual "SAVE" and "OLD" commands are used to store and load these programs.

The third form used by XB is the "merge format" stored in the a "DISPLAY VARIABLE 163" file. This form is created when the "MERGE" option is specified on the "SAVE" command and is loaded by the XB "MERGE" command. The beauty of merge format is that when it is loaded it does not necessarily overwrite the program in memory. The "MERGE" command does just that - it merges the new program (or program segment) with the program in memory according to the line numbers.

Now, we go to the good stuff, Assembler language programs. There are three forms for an assembler program: tagged object, compressed tagged object, and memory image. Tagged object is stored in a "DISPLAY FIXED 80" file on disk only. All program data is in hexadecimal so that it can be edited by the E/A editor. Tagged object can be loaded via "CALL LOAD" in XB, option 3 on the E/A menu, option 1 on the MM menu or by "CALL LOAD" in TI BASIC when either the E/A or MM

module is used. The program can be "absolute" or "relocatable". An absolute program must always be loaded at the same place in memory. A relocatable program can be loaded any place in memory. A tagged object program may have references to other programs or subroutines. The loader will resolve these external references, except for the XB loader.

Compressed tagged object is very nearly the same as tagged object except that the program data is stored as bytes rather than as hexadecimal digits. Compressed tagged object loads faster than regular tagged object as you would expect. The XB loader cannot load compressed object.

Tagged object, in either form, is produced by the Assembler when it assembles a source program.

The "memory image" form of assembler programs is the most compact and the fastest loading. It can be stored on cassette or disk. It is identified as "PROGRAM" in the disk catalog (just like a BASIC program). Memory image programs can be loaded by option 5 on the E/A menu or option 3 on the TI Writer menu (and I assume, by Multiplan, although I have never tried since I don't have Multiplan). It should be noted that there is one slight but important difference between how the E/A call a memory image program and how TI Writer does. TI Writer blanks the screen just before calling the program and the E/A does not. This means the program must turn the screen back on or nothing will show. Memory image programs are created by a Utility Program (one is provided on the E/A disk).

A PROGRAM file, containing an Assembler memory image or a BASIC program, can be read or written to any input/output device with a single I/O operation. This is one of the reasons they load so quickly.

There is a restriction on the size of an Assembler memory image program of 2400 bytes (9216 decimal). However, the E/A and TI Writer modules will load multiple memory image files to make a program on any size. They use the convention that the file name of the second and following files is obtained by incrementing the last digit or letter or the previous file name. For example, the TI Writer editor consists of two memory images files: EDITA1 and EDITA2. As a matter of interest, the ADVENTURE, Tunnels of Doom, Personal Record Keeping, Statistics and Personal Report Generator modules use a memory image or "PROGRAM" file

images can be saved or loaded with a single I/O operation makes them attractive for such uses.

A lot of Assembler language games that are circulated around are in the memory image format so let's look closer at them. Assembler memory image files have a three word header followed by the data to be placed in memory. The three header words are:

(1) This word is a "flag". If it is not zero (i.e., FFFF) then this file is not the last in a multi-file program. For example, the flag word EDITA1 is FFFF indicating that there is another file called EDITA2; the flag word in EDITA2 file is 0000 indicating it is the last file and there is no EDITA3.

(2) This word is the length of the memory image bytes, including the six byte header.

(3) This word is the CPU memory address where the memory image is to be loaded.

Execution of a memory image program always begins at the first byte of the first segment loaded.

Finally, the seventh form of program. This form is created and loaded by "EASY BUG" of the Mini Memory module. It can be written only to cassette and is a memory image, but is slightly different from the E/A memory image file. The EASY BUG memory image program can consist of only one segment. The header on the EASY BUG format is two words, as follows:

(1) The word is the CPU memory address at which the memory image is to be loaded.

(2) This word is the length of the memory data, not including the four header bytes.

If this whole thing is too complicated - maybe a table showing all options will help.

FILE TYPE	CONTENTS	MODULE	DSK	CS
PROGRAM	BASIC Program	Console	YES	YES
PROGRAM	BASIC Program	XB	YES	YES
INTERNAL V 254	BASIC Program	XB	YES	NO
DISPLAY V 163	MERGE Program	XB	YES	NO
DISPLAY F 80	Tagged Object	XB	YES	NO
DISPLAY F 80	Tagged Object	E/A	YES	NO
DISPLAY F 80	Tagged Object	MM	YES	NO
DISPLAY F 80	Compressed Object	E/A	YES	NO
DISPLAY F 80	Compressed Object	MM	YES	NO
PROGRAM	E/A Memory Image	E/A	YES	YES
PROGRAM	E/A Memory Image	TIV	YES	YES
PROGRAM	MM Memory Image	MM	NO	YES

Tips on Using Compuserve
 from Long Island 99'er Users Group
 D. Shigley 75746,673

Based on my own experiences, and those reported by others, I have written this file to assist users, especially new ones, in using the IBM SIG, as well as any other sig which they may use. Most of us know the heady feeling that one has, mixed with a lot of confusion, when first discovering Compuserve and the various special interest groups (SIG). We tend to forget, especially at only 300 baud, that the meter is ticking minute by minute for as long as we are anywhere within the system. The first month's billing comes as a shock which makes many decide to give up immediately in order to be able to eat and pay the mortgage, too. Thus, the suggestions given here are made to help you use this SIG yet avoid total bankruptcy also.

There are a few rules that you should learn, and adhere to, when you use Compuserve:

DOWNLOAD ALL MESSAGES, DO NOT READ ANY MESSAGES ON-LINE!!

This is probably the most important rule that you should learn since it will save you an enormous amount of time and money. Time will be saved since it is more efficient to read messages page by page on a screen, or even from a print-out, than to read line by line as the message appears. Money will be saved since you may reread, study, or ponder over whatever you wish without incurring more cost than that required for the download of messages.

A corollary of the above is to avoid fancy methods of scanning messages, marking them, and then reading or downloading them. You will find that such methods take more time than merely downloading all messages which have been posted since the last time you have accessed the board. You should also set your board options so that all messages will scroll across your screen and into your disk file without having to hit a carriage return after each one. This saves a great deal of time and, naturally, money, too. To change to this option, at the command prompt or from your menu, type the following:
 op;ns;p;t.

Note that when you are running in the non-stop (ns) mode, you cannot skip messages using Control P since some messages will be lost while your command is reaching

Compuserve. A further advantage to using the non-stop mode is that you will be less tempted to study or mull over a message: you won't have much time to do so, especially if you run at 1200 baud. Wait until you can read those messages on your own time, and free, from your downloaded file!

Another tip. Don't bother first downloading the messages marked for you. This is a waste of time and money. They will be in your message stream anyway, and since they might be of interest to others, you shouldn't delete them after reading unless they are just a thank you, an acknowledgement, or the type of cryptic reply I often make.

You should also start using the so-called Brief mode as soon as possible. This mode suppresses the menus, saving time and money as well as making your message downloads easier to read. To change to the Brief mode type the following from the menu:
 op;br;p;t.

As you have already probably learned, using the 'rn' command will make all the messages which accumulated since the previous time you were on the board be sent to you in the order which they were received. The weakness of this is that messages on the same subject will be scattered throughout your download. So, you should use the option of reading by thread to download messages. In this way you will see messages about each subject grouped together. To read messages this way type 'rtn'. The 't' stands for threaded. In other words, the messages with a common subject are 'sewn' together.

However, if your system or telephone line is flaky, you should stick to the chronological order for downloading. Otherwise, it is almost impossible to determine what messages were not downloaded. Then, it is better to rely on chronological order since it is easier to merely ask the system to send messages from the one next after the last one received using the 'rf' command. Remember that since you were cut off, the counter of the messages you have received had Not been updated because you hadn't logged out of the sig normally.

COMPOSE MESSAGES REPLIES OFF LINE AND UPLOAD THEM

This is really more of a corollary to the first rule. Using a decent text editor, one which can save messages in ascii format, i.e. without codes in the text, compose your

messages and replies in a single file and upload it when you log on the board. an example of how such a file might look follows:

re 47777

This is the answer to message number 47777
 -er /ex
 s re 48977
 Thank you for help. -er
 /ex
 sp
 re 48978

Note that the above message was saved as a private message since its phatic content is of no interest to others. /ex

s
 re 53201

This is an example of how you should save a message so it retains the format you wish. The 'su' takes care of it.

Name	Cost
first	32.50
second	12.90
/ex	
su	
1	

earle robinson 70135,141
 caribou in Florida
 Why do you spout such nonsense. -er

/ex
 so

*note- TIPRO users would end msg with s8 if using 1 option. Also, when using 1 you must use name and ID# if for a particular person, or ALL . The next line then is subject of msg. **

If you don't own a text editor, I recommend that you purchase one. IBM's Personal Editor is quite good, though Kedit, similar but much better, is what you should use. It is easy to learn and the price, the same as IBM's is reasonable: \$95. You may use your word processor to compose messages if, and only if, it is possible to create messages without control codes. If your processor can't do this, you need a text editor.

*note -For the TIPRO I highly recommend DF-EDIT and it is only \$50.00 Info on how to order is elsewhere in DL8.

You may have noted that the above examples use the 're' option to answer messages. Use of it will save a couple of lines and some seconds of log-on time.

I demonstrated messages using the FILGE editor. The default editor provided by Compuserve, called SED, should be replaced with filge since the latter will run faster and doesn't require special formatting codes when you wish to format a message. To change to filge, type the following at the function or menu prompt: op;fil;p;t. The 'p' makes the option permanent and the 't' returns you to the prompt or menu level. The '/ex' at the end of each message is the virtually the only instruction you will need to use the filge editor. It tells Compuserve this is the end of the message: the following line tells how to save that message.

You may also have noted that the above messages were nearly all sent using the RE(ply) option. This is a neater and shorter way of preparing a messages. It saves a couple of lines, too. However, be careful that you use the right message number to which you are replying and that it hasn't scrolled off before you upload your new message. Obviously, on a busy board, you will not be able to reply to a message which you saw a week earlier; it will have scrolled off and you will get an error message when you attempt to upload the message file for that particular message, requiring your braking off the sending of the remaining messages.

AVOID CONGESTED PERIODS ON COMPUSERVE

You may already have noted that there are periods when throughput seems very slow, while at other times it can zip along. Try to pick times to access Compuserve when throughput is at its optimun. This will save you time and your patience, too. The best times are very late at night or very early in the morning. If you do note sluggish throughput, and especially if you get one of those dreaded "Files Busy...." messages which could make you wait for as long as 15 minutes, get off and try again later. The worst times to use Compuserve seem to be Monday and Sunday Evening, though other evenings during the week can be congested, too. Slow throughput can result from your local node, and not just from the host mainframe of Compuserve. Note that if you are using a 1200 baud modem the extra cost of accessing during so-called prime time is

quite moderate, 4 cents per minute. So, under 1200 baud don't hesitate to use CompuServe during the day, if you can, since there is usually less congestion. Personally, I find the best time to access is early in the morning. Since I download messages at that time automatically I can look at them later in the day or during the evening when I have the time.

PLAN ON BUYING A 1200 BAUD MODEM

If you do intend to use time sharing services with any regularity, get a 1200 baud modem. It will pay for itself in a short time. On CompuServe the cost for 1200 baud is only twice the cost for 300 baud, and though you never get a full 1200 baud speed, you will more than pay for the difference in cost. Some timesharing services have no surcharge for the higher speed either.

Be careful in selecting a modem! Some have poor filtering against noise and will poorly filter out so-called 'noise' on your telephone line. Choose a modem about which you have had heard positive comments!

Don't be the first 'on your block' with some new brand: It may be a most costly decision!

Another thought on modems, regards the choice between an external one or a modem card to insert in your computer. Though the modems are often cheaper, I recommend buying an external one for two reasons. First, it can always be used on another system: you are not tied to the original computer for which you bought the modem. Secondly, the display lights on the front of most modems are useful in checking that everything is proceeding as it should.

SAVING (OR STORING) MESSAGES. This is sometimes confusing so here are a few pointers and reminders.

1. All new messages, those left with the L(eave) command must have a section number appended. Examples: s0 s2 su0 sp3

2. To have a message maintain the format you created, such as indents and spaces between paragraphs, use the 'u' in your save instruction. Examples: su sul spu3

3. If private messages are enabled, use them for things which are private, like leaving your lover's telephone number, or of a nature to embarrass the recipient such as a rebuke of his remarks. Most importantly,

private messages are ideal to leave a brief thank you to someone for service rendered. Be sure and delete private messages after you have read them to free up message space!

4. Try to store a message in the section of your sig which best suits its content. If it is about databases, and there is such a section, use that one for saving the message.

A FEW MORE TIPS. CompuServe messages are limited to 79 characters. Set your margins to 79 characters for messages you compose. The reason for 79 characters rather than 80 is because of the carriage return added by the system at the end of the line will insert an extra blank line. Messages on a SIG are limited to about 2500 characters, approximately 24 or 25 lines of text if you use 79 characters maximum per line. If your message is longer, split it into two and send them one after another. Use the same REply number for both.

*note - The TI sig now will accept 80 character lines.

Never format your messages with the right margin justified! Those with margins different than yours will see poorly formatted messages, and the extra spaces take up time and space, costly for you and your recipients.

Don't get fancy in formatting messages! They are more costly for everyone who sees them. Some people have the annoying habit of formatting a new line at the end of their messages and putting their signature at the right side of the line. Others say 'Have a nice day' at the end. This is useless and merely makes more money for CompuServe at your expense. It is also perfectly useless to add your ppn (for the curious, personal project number) to the end of a message; It is already in the message heading.

Above all, don't use asterisks or other gimmicks around messages! Not only do they make reading slower, but they are ugly, especially for users who may read them on the screen with a different line width than yours. If you have not enabled lower case, do so immediately by running the 'Default' program. From the function prompt on the sig, type 'g cis-9'. If you yourself prefer reading messages in caps, you can even set your parameters so that all messages you input are in upper-lower case and all messages you download are in caps. 'Duit'

yourself but please do not force others to read them that way!

*comment - I think the use of any "gimmick" is ok if appropriate to the msg or if it helps make a point. Do be aware that other BB'S and sigs use different line widths and your msg will not be formatted the same way you typed it.

*note - HI DON.

I hope that anyone reading this may find some assistance in better using Compuserve. It is a pity that many new users are rapidly discouraged due in large part to the first monthly bill and do not realize that it is possible to economically use Compuserve if they only organize their time. As I said above, the most wasteful practice is indeed reading and answering while on the system while the meter is ticking away. This is as silly as if you were on a long distance telephone call and told the party that you had to go take a shower and that he should hold on since it 'will only take a few minutes'.

*comment - Somehow I've lost the heading to this article that contained the heading and author's name but assume it's from the IBM sig contributed by Earle Robinson. Tried to find it again on CIS using key words to no avail but in any event, Thanks Earle!! Your suggestions have greatly enhanced my use and enjoyment of Compuserve and I'm passing them along to the folks at TIsig.

SPEED:

There is a wide diversity in the speed of various computers. The execution of a command on a small computer may be measured in less than a millisecond, a one-thousandth of a second. Many computers execute an instruction in microseconds, one millionth of a second. The modern supercomputers are approaching the nanosecond range. How fast is a nanosecond?

If a nanosecond were one mile>>>
then 1 second would be 2000 trips to the moon and back.

If a nanosecond were one minute..
then a second would be 1900 years!

MINI MEMORY TIPS

Courtesy TI SOURCE (tm) BBS

While playing around with the Minimen module, made some interesting finds in Expansion memory called EXPMEM2. It is well known that programs can be saved to the battery-backed RAM in the module, programs can also be saved to EXPMEM2, freeing the V RAM for other programs. This can be handy if you are doing a lot of work in TI Basic, but would like to have something like a disk directory program handy. Just load your directory program, do a SAVE EXPMEM2, and you can then type NEW or anything else and your program will stay tucked away in the Memory Expansion. When you want to use it, you can go get it with, OLD EXPMEM2, and then perform a RUN (no, it cannot be MERGED with a program already in memory). How stable is this program? Here is what I did to find out; I entered a program into EXPMEM2, executed a BYE, removed my Minimen, Inserted the Extended BASIC mod and when it was running typed CALL INIT (wipes out everything in memory, right?). I then exited XBasic, went back to Minimen and typed OLD EXPMEM2, and (you guessed it) the program was still there!

There is also another location called EXPMEM1, apparently located in low memory at >2000 (more tech talk), and completely undocumented in the Minimen manual! It looks as though it might be possible to have files open in the Minimen RAM, and in EXPMEMs 1 and 2 at the same time. Or, how about two different Basic programs, one in E1 and the other in E2.

One last Minimen trick, this one by way of Miller's Graphics. You can use the "save to EXPMEM2" trick to run cassette programs that will not run because your disk system ties up V RAM. Here's how:

First, enter the program from tape (everyone probably has a few of these that you never load anymore because you know when you type RUN you're going to get "MEMORY FULL IN XXX"). Next type "SAVE EXPMEM2". Now type CALL LOAD(-31888,63,255), which "disables" your disk drives and gives you back the memory they took (you DID remember to plug in your Minimen, didn't you?), and type NEW. last, type OLD EXPMEM2, and like magic your unRUNable program moves into the expanded VPD and will run. Nifty, no?

HOW TO DOWNLOAD XMODEM FILES FROM COMPUSERVE WITH FAST-TERM

by Barry Boland DVUG

Due to the newsletter space limitations, this article will only cover XMODEM binary (8-bit) files and Fast-term. If response is favorable, I'll do more articles later on ASCII and TE2 protocol transfers.

The first thing you need to find is the file you want to Download. Log on to CompuServe, and go to the TI-Forum (60 TEX200), then at the "Function:" prompt, go to the DL you want to search. (type DL1 for this example). You'll find yourself at a new prompt, "DL 1:"...at this point you have 2 choices:

A: type 's' (for Scan)...this will list out the NAMES of ALL the files in that DL section (WARNING: this can be a long list!).

B: type BRO (for BROWse)...this will step through the list of files one at a time, giving you the CIS ID* of the person who uploaded the file, the CIS NAME of the file, the date the file was uploaded, the "size" of the file {xmodem uploads have 2 numbers here, for example: 21810(9312)}, the number of times that file has been "accessed" (downloaded or read), a list of KEYWORDS, and a short description of what the file is/does.

I generally use the BRO command, after each file listing, there will be another prompt, "(R D T):"...here you can Read, Download, or return to Top. The T (for Top) will return you to the "DL 1:" prompt, where you can type T again to return to the "Function:" prompt of the TI-FORUM. Read will "read" the file...this ONLY makes sense if it's an ASCII (or TEXT) file. look for the KEYWORDS "ASCII" or "TEXT" as a clue. If this file is not what you want, you can press <ENTER> on your keyboard, and you will be stepped down the list to the next file with the "(R D T):" prompt again at the end of that file.

Let's say we've found the file we want. Type D (for Download). You'll see a list of protocol choices...what we want here is 1 XMODEM (modem7), type 1. You should now see the msg:

Starting XMODEM transfer

Enter a carriage return when transfer is complete.

Now we want to switch Fast-Term into its XMODEM sub-routine. Have you named your file yet? If not, right now press "FCIN N" and type in the name you want the D/L'ed file to have ON YOUR DISK (remember NOT to use any "."'s here, Your disk system won't read them). File already named? Good, on to the next step. Press, and hold down the "FCIN", "SHIFT", and "X" keys... you want to have All 3 pressed down at once. This puts you into XMODEM mode. You now have to tell Fast-Term whether you want to Send or Receive. We want to Receive, so press R. The next prompt is for CRC, checking, CompuServe supports CRC, so enter Y for Yes.

FINALLY, you can now sit back and watch the file "stream" into the XMODEM buffer, and every 41 records it will stop momentarily to "dump" the buffer to your disk. When the TRANSFER COMPLETED msg comes up, just press the <ENTER> key, to tell CIS that you're done on your end.

Now you'll be back to the "(R D T):" prompt at the end of the file you just DOWNLOAD. You can go down the list and look for another file to download (you won't get the XMODEM (MODEM7) prompt the 2nd time, it "remembers" that choice, otherwise its all the same again). OR, just type in 'T' here to get back to the "DL 1:" prompt. If you want to look at one of the 6 other DL sections, this is where you type the one you want. For example, to switch to DL4, just type DL4 at the "DL 1:" prompt.

All done? Type 'T' at the "DL 1:" prompt to go back to the "Function:" prompt of the TI-FORUM. You can now type in RN (for Read New msgs), or just type OFF to log off CompuServe.

If you have any problems or questions, I can be reached through D.V.U.G> TIBBS (322-3999) or right on the TI-FORUM. My CIS ID* is [74756,176]

PEEK AND POKES AGAIN
INSIDE THE TI 99/4A by Et.Al.

"USEFUL" XBASIC CALL LOADS

CALL PEEK(2,A,B):CALL LOAD(-31804,A,B).
Same as using the command "BYE".

CALL LOAD(-31961,51):: END.....
Returns you to the title screen (with full graphics)

CALL LOAD(-32630,128).....
Returns you to the title screen (with-out graphics)

CALL PEEK(-28672,A).....
This checks to see if the speech syn. is attached. (Great for optional speech programs) If the syn. is attached Variable A returns a value of 96, if not attached, 0.

CALL LOAD(-32699,2).....
Activates ON WARNING NEXT

CALL LOAD(-32699,4).....
Activates ON WARNING STOP

CALL LOAD(/32699,16).....
Activates TRACE

CALL LOAD(-32699,64).....
Activates ON BREAK NEXT

CALL LOAD(-31888,63,255).....
Type in this and then NEW to shut down your disk drives for those extra long basic programs to load in. see the next CALL LOAD to turn them back on.

CALL LOAD(-31888,55,215).....
This when used with a CALL INIT first will do the oppposite of the above CALL LOAD.

CALL LOAD9(-32699,0) OR (-31931,0).....
Deletes Extended Basic protection

CALL LOAD(-31931,128).....
Installs Extended Basic Protection

CALL PEEK(-31863,A).....
"A" will equal 231 if 32K is present.

CALL PEEK(-31952,A,B).....
Is the pointer to starting address on line number table, 4 bytes per entry, (2 for number line,2 for start addr.)

CALL LOAD(-32729,0).....
This loads any program in disk one called "LOAD"

CALL LOAD(-31961,149):: END (OR STOP)...
Will reset the console and search for a filename on disk one called "LOAD"

CALL PEEK(-31950,A,B).....
Is pointer to the ending address of the number line tables.

CALL PEEK(-31954,A,B).....
Is the current line being referenced in the table.

CALL LOAD(-31888,63,255).....
This will not reserve any room in the VDP RAM for Disk Buffers.

CALL LOAD(-31806,16).....
THIS WILL DISABLE THE FCIN QUIT key.

CALL LOAD(-31868,0).....
When within the body of a program, and when FTCN 4 (CLEAR) is pressed, Listing the program is impossible.

CALL LOAD(-31878,X or (-31806,X).....
Makes all sprites >X stop
XBASIC CALL LOADS TO PLAY AROUND WITH

CALL LOAD(-31740,A,B).....
A & B=Values you enter. Change the around to get different sounds. They will stay on until another sound is made (input and error beeps.)

CALL LOAD(-31748,X).....
If you make X=0 then all tones stop, and the cursor halts. X can equal from 0 to 18 with different results.

FUNNEL WRITER

A PROGRAM FOR ALL SEASONS

Joe Nuvalini - Front Ranger U. G.

I recently received an exceptional Fairware disk from two gentlemen in the Hunter Valley 99 Users Group in New South Wales, Australia. Their names are Will and Tony McGovern and the disk contained version 3.3 of Funlwriter which, according to an earlier letter I received from Tony, will be the definitive version of this program.

This is by far the most versatile program I have seen for the 99/4A. It allows you to use TI-Writer(TIW) and the Editor/Assembler (E/A) without their respective modules. The program auto loads from Extended Basic(XB) and will also load from TI-Writer, the Editor Assembler or the Mini-Memory module. The disk contains TI-Writer, the Editor Assembler, Disk Manager 1000 VER 3.1 (a FREWARE program of the Ottawa User's group), a sector editor, and a FORTH loader. You can also load your assembly programs without the use of any modules except Extended Basic. To run the disk you need the console, 32K memory, and the disk controller and drive. It also helps to have a second drive and a printer.

There is a file called -READ ME- and six FWD0C files that should all be printed using the TIW Formatter and- more important - read before you begin. When you're done with that then copy the Funlwriter disk so you have a working copy and put the back-up copy away in a safe place.

Before loading the program examine the LOAD program. LINE 120 allows you to set the primary and alternate screen colors. LINES 130 and 140 set the default options for the PF option of the Editor(130) and the Formatter(140). LINES 160 through 190 allow you to enter the names of programs you want on the User's List option while lines 240 through 280 are the load commands for these options. You can set a value for K in LINE 210 that will be the default for the drive number that appears on the screen with DSK. DO NOT RESequence the program or you will destroy the LOAD program. The FWD0C/LOAD file explains how to set up the User's List options and the various methods of loading Funlwriter.

Now select a method and let's load the program. The first thing you'll see is the title screen followed by the first menu with

three selections: TI-Writer, Edit/Assm, and User's List. We'll cover option three, User's List later in this review. If you select option 1 or 2 you arrive at the central menu which has 6 selections. They are Editor, Formatter or Assembler, DM1000, Utility, Switch, and Reset. Selecting Switch changes option 2 to Assembler, c-Compiler and back to Formatter so you can switch between these functions. I might mention here that the files C99B through C99E will load REL2 of the c-Compiler by Clint Pulley. It loads from this menu using file C99B through E. This is the preferred method of entry. It may also be loaded from the program file loader, discussed later, by entering C99C at the filename prompt. You must have the rest of Clint Pulley's small-c files for this option to be of any use to the user. Pressing Reset places the current filename you have been working on into the mailbox so that if you leave TIW or E/A and go to another Funlwriter function, say DM1000, and then return to TIW, when you select Editor or Formatter that filename will be there for you to load or print. After selecting Reset, the option six name changes to Quit, and pressing that option returns you to the Master Title Screen. We'll discuss option 4, Utility after we finish our discussion of TIW and E/A.

If you select option TI-Writer from the first menu you then can select The Editor or Formatter from the central menu. The Editor functions like the TIW editor with the three following improvements:

1. If the loader can find a filename in the mailbox it writes it into the LF/SF buffer, which otherwise shows DSKx. when calledup with x being the default disk drive set in the LOAD program.
2. The quit function remains disabled at all times while in the Editor.
3. The Show Directory(SD) function is an assembly routine that allows single key paging through the files. Fractured files are indicated by an asterisk after the file length.

The Formatter is the same as TIW with the following improvements:

1. There is now a Quick directory(QD) function here any menu in the program. To access it, enter FCIN 7 (AID). It operates the same as the SD function in the Editor.
2. The Formatter will automatically display the last file used when it can find one. If no name is present then DSKx. appears.
3. The FCIN 9 key allows you to return to the Funlwriter central menu.

If you need to reload either the Editor or Formatter immediately after exiting then they do not need to reload from disk.

If you select the Editor when Assembler or c-Compiler is in the second position of the central menu, a modified version of the TIW Editor is loaded which is suitable for use as a source code Editor. Word wrap is disabled, E/A tab defaults are set, and no final tab record is written to disk. To write a DF80 file to disk you use the PF option using F dskxetc as described on the TIW manual. The Source Editor loads CHARA2 which is slightly different than the CHARA1 file that is loaded by the TIW Editor. This acts as a reminder to let you know which editor you are in.

The Assembler has some enhancements added. The filenames are visible on the screen while it is executing. You can use AID to give you QD allowing you to check the filename on the disk. If a filename is found in the mailbox it is written as the source code filename and the object code is the same name with the last two characters removed. Also R is automatically entered in the Options field of the Assembler as default value.

Utility, option 4 on the central menu, brings you an assortment of assembly file loaders called the Program Load Environment (PLE). This menu displays five options on the screen but has a total of * options, the last three of which are entered in the blind. Option 1 is for loading TIW utility files like Dragonslayer's Spellchecker. Option 2 sets up a GPL environment for loading other self contained program image files while option 3 is the E/A "RUN PROGRAM FILE" function. It should be noted here that the program file loaders will support cassette files by entering "CS1." (see E/A manual for more information on this function). Option 4 is the E/A "LOAD-RUN" function and handles object files, compressed or not, and even displays the DEF table so you don't have to remember the program execution name if it doesn't auto start. Option 5 is RE-ENTER (1-3) and it allows immediate re-entry to a program without reloading it, assuming it is re-enterable. The invisible options (6,7,8,) allow other object code loading options but in the interest of brevity I will not go into them here. Information on these options can be found in the FWDOC-EASM file. Entering FCIN 9 from this menu returns you to the central menu.

Now we'll discuss the User's List option option 3 on the first menu. This menu has options. The first 8 options can be used defined although the LOAD program comes to run DM, the yarc disk manager as option 6, Dpatch, the TI sector editor DISKO as option 7, and a TI-Forth loader as option 8. Option 9 is BACK and it will return you to the Funlwriter title screen. This menu is seen in the LOAD program, as are the loaders. You can run XB program or object files from this menu if the corresponding files are placed on the Funlwriter disk. The XB programs are called by a RUN "DSK1.filename" statement. The E/A files are loaded using a CALL LINK("UTIL",filename,K). The numeric parameter K is the same as would be entered from the PLE discussed earlier, i.e., 3 for a E/A program file and 4 for a object code file. I find this part of the program particularly useful as you can put your favorite utility programs on the Funlwriter disk and have them available. In addition to TIW and E/A I have Masscopy, Fast Term, 4A/Talk, PRBASE, DM1000, DPatch, the TI-Forth loader, and a program called Recall, all available through Funlwriter. I should mention that I am using a double sided disk to hold all of that. You will be somewhat restricted as to what you can put on a single sided disk with the Funlwriter files.

There are several other files supplied with Funlwriter that deserve mention here. FWSAVE utility is for us with E/A to allow SAVEing of any program loaded as an object file by Funlwriter into low memory. UPATCH is a patch that creates a file called UTIL1 once you have your screen color and printer default set in the LOAD program. UTIL1 is used to re-enter Funlwriter from several areas. APATCH file is used to modify the ASSM@ file from your original E/A disk to work with Funlwriter. The ASSM file is created in 22 sectors long, 2 sectors longer than the original. It appears that the authors have already done this on the disk provided. FWRMM is for us with the Minimemory module to load the UTIL1 file into high memory.

A word here about Fairware... Will and Tony have set no price for this program but merely say "I can suggest only that you judge the program on it's own intrinsic merits, perhaps measuring its worth by how much you use it as compared to other "fairware" or commercial programs that you use. I might suggest you do what our user's group, The Front Rangers, did. We collected donations from the members of our users

group who wanted the program and sent one international money order from the club to the authors. Be sure to include Two disks when ordering your copy unless you have double sided capability as the DOC files are over 200 sectors long. Also be sure to enclose a couple of dollars postage as mail to and from Austrailia is not cheap! This is truly a fine piece of software. Lets make sure the authors are adeqately compensated for their work.

Authors: Tony and Will McGovern 215 Grinsell St Kotara, NSW 2289 AUSTRALIA

HARNESSING THE POWER OF SPEECH

Originally by Craig Dunn of the Central Texas '99/4A Users Group February 1986, via the DELAWARE VALLEY USERS GROUP.

The TI Speech Synthesizer is an amazing little device. It was a breakthrough for the lower end (priced) computers. Unfortunately, many 99/4A owners still don't know how to access speech along with all its little features. Sure, a lot of games use speech to add interest and excitement but the applications of speech goes far beyond games.

One of the major features of the speech synthesizer is its ability to let you add speech to your programs. There are several ways to do this, including TI's Terminal Emulator II, TI's Extended BASIC and through assembly language routines. Extended BASIC provides a rather limited vocabulary (unless you are using one of several recent utilities that give you unlimited speech in Extended BASIC, but that's another story). Terminal Emulator II (TE2) allows for unlimited speech directly from BASIC. This built-in-text-to-speech capability of TE2 will be the focus of this article.

First plug in the TE2 command module, turn on the computer and select TI BASIC. Now type and run the following program:

```
100 OPEN #1:"SPEECH",OUTPUT
110 INPUT A$
120 PRINT #1:A$
130 GOTO 110
```

If you get an error, make sure you have the speech synthesizer connected properly to the side port. Now we have a very simple text-to-speech editor. Line 100 contains the OPEN command needed to access TE2 speech capabilities. Line 120 sends the text strings that you type in to the

text-to-speech interpreter, which then sends the info to the synthesizer. Experiment with this for awhile by typing in phrases, followed by an enter.

In the above example, you were in the default speech mode. This means that no commands have been sent to alter the voice. We can change the voice easily using the "/" command. The proper format is:

```
//PITCH SLOPE Ex: //34 118
```

The PITCH is a number between 0 and 63. A zero causes the speech synthesizer to wisper phrases. Pitches from 1 to 63 range from the highest pitched (1) to the lowest pitched (63). For best sound, figure the SLOPE using the following formula: SLOPE=32 times (PITCH/10).

Round this result to the nearest whole number. Now when you enter the command along with these two numbers, it will appear that nothing has happened; But, type in a simple phrase and press enter. You'll notice the change in voice. For example, at the prompt in our simple little speech editor, type "//55 176" and press enter. (Be sure to include a space between numbers.) Nothing happened, right? Well, now type something in and press enter. See how the voice changed? It became much deeper. Now try "//0 0" and press enter. Again, type in a short phrase. Another voice tone! Experiment with these and other PITCH/SLOPE combinations to get the feel of the working with these.

Before we wrap up this tutorial, we'll take a look at the inflection symbols. The symbols are "^" (caret), "_" (underline), and ">" (greater than symbol). The "^", when placed in front of a word, indicates a primary stress point to the text-to-speech interpreter. Only one "^" may be used per string. The "_" is used to indicate a secondary stress point and may be used without limit through a string. The ">" will shift the stress points within a word. Experiment with all these to make words sound better and more human-like. Remember, all inflection symbols must PRECEDE the word they are to affect.

Will I hope someone benefitted from this article. On a final note, remember that the text-to-speech interpreter isn't perfect! Sometimes you might have to alter a words spelling drastically to make it sound right. Have fun!

NEW HORIZONS RAMDISK

Gary Tanner SAN ANTONIO 99ers UG

I have put off buying a RAMDISK for quite some time, only because I wanted to be sure of two things. First and foremost it had to be a dependable product and secondly a flexible one. Enter HORIZON COMPUTERS LIMITED new RAMDISK. It filled both objectives and it also has a battery backup feature for retention of data.

HORIZONS COMPUTERS LIMITED should not be new to you. Originally they produced the TI-COMM bbs that was distributed to UG around the country. This was a bare bones bbs program of mediocre success. The next venture for them was what now has become known as the SUPER CART. If you are unfamiliar with this, it allowed people to add 8k of Ram to the E/A cart. Their latest product, which may prove to be their greatest contribution to the TI-COMMUNITY, is a battery backed RAMDISK. It is available in two versions, SSSD OR DSSD. The RAMDISK has 104K for the SSSD AND 192K for the DSSD. The operating system for the RAMDISK is in Ram so that along with the card you receive several disks, one of which contains the operating system. The program for the operating system takes up approx. 8K on the card with some extra space for future modification of the operating system (an extra 4K of ram). So the card comes as a 90K unit for the SSSD and 180K for the DSSD card.

The assembled card comes as a ready to use item. That is, all you have to do is open the Expansion box and install the card. Load the operating system and copy the files you use most and you are ready to go.

The card uses low power CMOS STATIC RAM CHIPS. Functions exactly like a TI floppy except operates at RAM speed. So flexible that it can be carried from computer to computer like a floppy disk.

Compatible with any software that uses a standard TI DSRLNK. (This means that there is some software that will not work with the disk.) The Ramdisk appears to be compatible with TI-Writer, Editor-assembler, Multiplan, Logo, Forth, Basic, EXT Basic, and Assembly Language.

However, some software designed to directly access the Disk Controller Card will not recognize the RAMDISK and hence is not compatible. Examples include the CORCOMP

Disk Manager and TI Pascal. Non standard disk access technique could also cause compatibility problems (ex., TI-ARTIST)

The only battery backed Ramdisk for the 4/A, Ni-cad batteries charge while the computer is running.

Supplied with the DM1000 (version 3.1 as the resident disk manager program, the DM1000 can be loaded via a CALL statement from BASIC in less than 2 seconds. Other call statements from BASIC are:

- 1) set the RAMDISK drive number (CALL DN(x))
- 2) set the max number of sectors (CALL MS(XXX))
- 3) set the write protection, (CALL WO)
- 4) turn on CRU for direct DSR access (CALL CO)
- 5) execute machine code from BASIC (CALL EX(adr))

DIP switch setting allows CRU addressing from >1000 to >1700. The Ramdisk may be named DSK1 TO DSK6 and comes with the complete DSR source code, (included with all Assembled units) including a separate manual that details all DSR routines. The included documentation explains how users may add their own assembly language call routines. It comes with development software, including a loader for E/A opt. 3. Allows object files to be loaded for modifying the DSR as desired.

The Ramdisk is available as a ready-to-run card (sssd or dssd) with 90 day warranty on parts and labor, or as a bare printed circuit board with parts list and instructions. All packages include the operating system with full code.

Enough of the technical stuff. How does it work?

Well, to date one word describes the whole setup. F A N T A S T I C. I have set up the FUNNELWRITER program on the disk along with FASTTERM, PRBASE, CATBIN, and a MAILLIST program. With the FUNNELWRITER present on the disk it gives me the power for computing what I want. How reliable is the product? Well I have had the RAMDISK for about two months now and so far have had no problems. Actually it is the most unnoticed item that I have in my system. That is, it just performs absolutely flawlessly.

People may wonder, considering the new products on the market, if I would still consider this a good product. The answer is "YES". The most liked feature about this product is that the whole system is in RAM. If you want to change the operating system, all of the source code is provided so that you can. Already there are people out there writing source code that will allow the card to act as an 232 card as well as PRINT SPOOLER, and even as a PSEUDO DISK CONTROLLER. The changes mentioned above are quite detailed and require an advanced knowledge of assembly language.

The only changes that I would make to the card would be for an easier way to add different CALL statements to the operating system. For instance you could add all the call statements for the files that you have on the disk and would simply do the appropriate CALL from Basic. The limitations of the card are that it is only available as a single density item and the maximum number of sectors available is 720, so you probably could not have all the programs that you use on the card at the same time.

Plans for the RAMDISK are to modify the operating system to allow more CALL statements, and also to build a second RAMDISK to add to my system and partition it so that it will be available as a print spooler and have the addition of another ss/sd drive.

But then that is another story, so until next time. further information concerning the RAMDISK can be obtained from me or by writing to:

HORIZON COMPUTERS, LIMITED
P.O. BOX 554, WALBRIDGE, OHIO 43465

PRICE LIST
ASSEMBLED UNIT

SS/SD (90K)...\$180.00

DS/SD (180K)...\$210.00

KIT PRICE

BARE PRINTED CIRCUIT BOARD..\$42.00

PARTS

KIT (SS/SD).....\$72.00

(DS/SD).....\$105.00

A REVIEW OF PRBASE

By Roy Long - Shoals 99'er (Edited)

A major deficiency in the available software for the TI is the lack of a database program that is first, simple and second, quick. WILLIAM WARREN of Aurora, Colorado satisfied this desire for the full utilization of my "TI99/4A" (the best home computer in the world.)

"PRBASE" is a "freeware" offering that is certainly worth every penny of the \$10.00 that every user should sent to William at 2373 Ironton Street, Aurora, Colorado 80010. By the way, I do not, repeat, do not receive any of the \$10.00, nor do I know Mr. Warren. Again, my interest is to have the best database program available to the TI world in order to utilize my own TI to its fullest. As you will see, later I even have one or two suggestions, requests, or observations to improve (better suit my needs) the program that I hope Mr. Warren can accommodate in another or later version of "PRBASE".

(Editor's note: As noted in the September issue of Micropendium; several of the suggestions made in this article were addressed in the newest release of PRBASE.)

Lets begin our review of "PRBASE". The program and its documentation came to me on a single sided disk. The files you create are also to be kept on single sided disk. This is due to a unique initialization of the file disk, which we will get to later. The program runs in Editor/Assembler - assembly language or in Extended Basic. You must have the Editor/Assembler module or the Extended Basic module, the expanded memory card and a disk drive (two drives are, of course, better than one). I did not like (at first) the single sided disk and the single disk drive approach to the program. I was able to get the program to load with relative ease after some initial problems on my part, and then set up and ran the program in drive 1 and the file disk in drive 2.

The documentation begins with an overview of the entire "PRBASE" system. First you design your data screens. Then, you build up to five (5) customized Tabular Reports formats and a customized mailing label format. At this point you can add, edit or delete data to the program format that you created. Data can now be retrieved by three methods: Global disk search, Field search, and the super fast memory Index search. Data once

found may be output to the printer or disk file as screen dumps, tabular reports, or one-across mailing labels and even may be incorporated into other document with use of and Display Variable 80 text editor, such as, TI Writer or the Editor of the Editor/Assembler. Data may be sorted in seconds with the "S" command.

"PRBASE" uses a unique disk format that cannot be read by the Disk Manager Command module. A single sided, single density diskette normally contains 360 sectors. With "PRBASE", the first 10 sectors are dedicated to storing information on your data screen design, your tabular report format data, and information needed to print your mailing labels. The remaining 350 sectors are used for storing your data. "PRBASE" file diskettes can be copied with most disk copy programs, such as, "Massscopy". You should name your disk "PRBASE" before you download the "PRBASE" programs and files, as the program uses the name "PRBASE" to load the special characters, etc. You can place a copy of "Massscopy" on the program disk with the name "copy" and the "C" command will copy from the data or file diskette for backups.

This may seem complicated at first. There is one thing I failed to mention, until now. There is a tutorial in the documentation that is a step by step approach to create your data entry and retrieval screen, the tabular report formats and the label format. Just take your time and go through it and before you know it, you have created your first database ("PRBASE") file. Don't be afraid to try out the program. It works, is straight forward, and with practice, is also simple. Try the various colors for the foreground and background. Also, you can jazz up the screen with the characters at the bottom to outline or box in your titles and field names.

OH! I almost forgot. "PRBASE" comes with four(4) programs: "PRBASE"; "CREATE"; "CHAR" and "LABEL".

PRBASE is the main database manager program. You'll use it to add, edit, retrieve or process your data. You will have to enter the "PRBASE" program by option #5 on the Editor/Assembler screen and "DSK1.PRBASE" or "DSK1.PRBASE" in the Extended Basic mode.

Create is used to date screens, your tabular report formats and your custom mailing labels. This is the first thing you will do. In fact, you have to load "DSK1.CREATE" in order to get the "PRBASE" program to run the first time. (After competing the tutorials then you will run "DSK1.PRBASE" to actually enter, add, edit or process the data.)

CHAR is a program file containing the lower case and special graphics characters used by "PRBASE". You cannot run "CHAR".

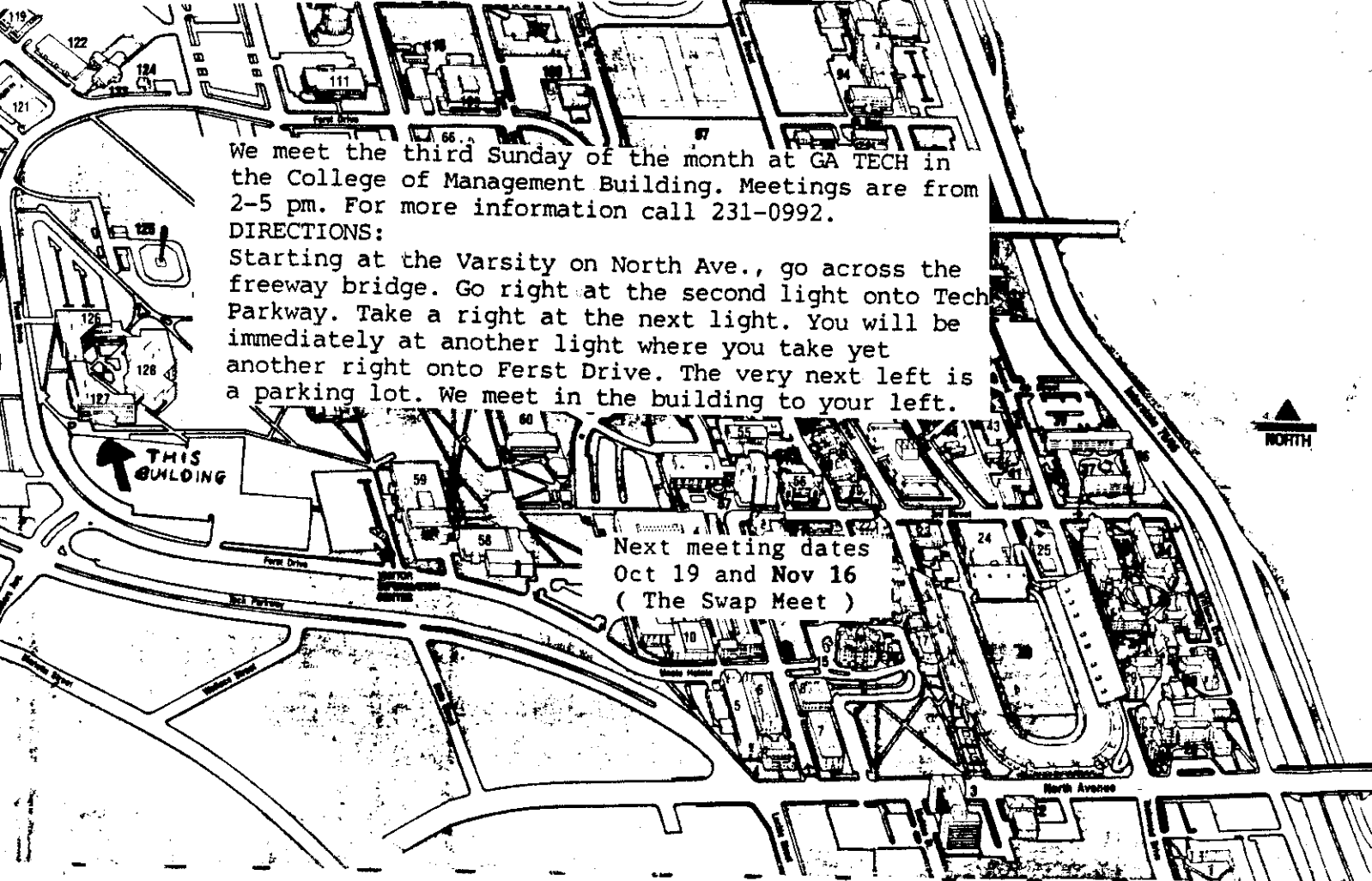
It is a TI Extended Basic program that prints bitimage labels to aid in distinguishing your "PRBASE" data diskettes from other TI diskettes.

After you create your data screen, tabular reports, and label formats you will next save it on another diskette. Note that you have to take the "PRBASE" program disk out of drive 1 and save on a single-sided, single-density disk in drive 1. Then, place the "PRBASE" program disk back into drive 1 and run "DSK1.PRBASE". You can then enter "DSK1.(your file name)" and the "PRBASE" program will load and run your data entry screen, tabular reports and label formats with full utilization of the two disk drives. Of course, you can operate the program with one disk drive by swaping the "PRBASE" and file diskettes in drive 1.

One of the main attributes of the "PRBASE" program is the "H" (HELP) command. It immediately places all the commands on the screen and at your fingertips. The program executes immediately and sorts and searches extremely fast. You can also create your own custom index to speed up searches in a field other than the first field which is automatically indexed and the Global search, which searches all the data entered.

By now, I think you understand that I am sold on the "PRBASE" database program. There are two observations that I have. I would like to be able to change the tabular reports, once they are created, without automatically erasing the reports already created. At least, I would like to select which report is to be modified or deleted. Also, I would like to have a field for handling money and a field for calculations. That would add, subtract, divide, multiply and calculate percentage, average, minimum and maximum entries. For now, I am sold on "PRBASE".

Again, I thank Mr. Warren for his efforts in assisting the TI world in having full use of the TI99/4A (THE BEST HOME COMPUTER IN THE WORLD)!



We meet the third Sunday of the month at GA TECH in the College of Management Building. Meetings are from 2-5 pm. For more information call 231-0992.

DIRECTIONS:

Starting at the Varsity on North Ave., go across the freeway bridge. Go right at the second light onto Tech Parkway. Take a right at the next light. You will be immediately at another light where you take yet another right onto Ferst Drive. The very next left is a parking lot. We meet in the building to your left.

Next meeting dates
Oct 19 and Nov 16
(The Swap Meet)

ATLANTA 99/4A COMPUTER USERS GROUP
POST OFFICE BOX 19841
ATLANTA, GEORGIA 30325

BULK RATE
U.S. POSTAGE
PAID
ATLANTA, GA
PERMIT No. 492

UG EX IN
Miami County Area 99/4A H.C.U.G.
P.O. Box 1194
Peru, IN 46970

* DUES ARE DUE THIS MONTH
** DUES WERE DUE LAST MONTH
*** THIS IS YOUR LAST NEWSLETTER