BITS,BYTES&PIXELS

LIMA 99/4A USERS GROUP

February 1994 — Volume 10, #2

JIM PETERSON
obituary

"THE TIGERCUB" died unexpectedly on January 12, 1994. He went out to get his morning newspaper, sat down in a chair to read, and never got up. Jim Peterson was known to many 99/4A users as a newsletter columnist (Tips from the Tigercub), a master extended basic programmer, and the organizer of a fantastic and inexpensive library of public domain 99/4A software. He usually attended the annual Lima MUG Conference and can be seen in the video tape record of many of these conferences. His most recent contribution to the TI community is a "Microreview" in the January 1994 issue of Micropendium.

Jim's family indicates that his business (Tigercub Software) will be closed and discussions are underway with his local user group (C.O.N.N.I.) concerning the business assets which consist of disks from his public domain catalog. The February issue of the C.O.N.N.I. newsletter will be a memorial issue in his memory.

PC99 - A FIRST REPORT
by Charles Good
Lima Ohio User Group

A growing number of 99/4A users also own IBM compatible computers. Such owners known to me include myself, Dave Szippl and some other local members of the Lima User Group, Bruce Harrison, Jim Peterson, Irwin Hott (runs the C.O.N.N.I. clearing house BBS), Jim Krych (of Asgard Peripherals), and Barry Traver. There are certainly many others. An obvious question for those who own both types of computers is, "Wouldn't it be nice if I could run some of my neat 99/4A software on my IBM compatible?" And for even those 99/4A owners who do not have their own personal IBM clone, "Wouldn't it be neat to run my favorite 99/4A software on my computer at work?"

I am writing this report using the 40 column Funnelweb v5 text editor running on my 386DX/40 IBM compatible computer. It will be printed right justified in condensed print with double strike in the appropriate places using the Funnelweb (TI Writer) formatter also running on my 386DX/40 system. PC99 is the magic that allows me to use this 99/4A software on my IBM clone. For years there has been talk of IBM emulation for TI users, maybe a new type of computer with two separate CPUs that would allow 99/4A and IBM software to run on the same machine. Something like this has now arrived! It doesn't use multiple CPUs and in fact it isn't even a new computer. The whole thing comes on a single 3.5 or 5.25 inch high density IBM compatible disk.

PC99 is a complete software emulation of an expanded 99/4A system designed to run from an IBM clone. Minimum requirements are a 386 or fancier (486, Pentium) processor, 640K memory, VGA graphics, and a hard drive. Disk size can be specified at the time of purchase. PC99 programs the clone's microprocessor to behave exactly as if it were an "all TI" expanded 99/4A system. This means the /4A console, 32K memory expansion, RS232/PIO interface, and TI disk controller with three DSSD drives on line. You can also purchase a version that behaves exactly like a 99/4 (without the "A"). All you do is install the software onto your PC's hard drive. Once installed, just enter C: PC99 PC99 from the C:\> prompt and your IBM clone will act EXACTLY like a 99/4A. The TI color bar title screen will appear on the VGA monitor, and when you "PUSH ANY KEY TO CONTINUE" you will get your choice of TI BASIC and any other TI cartridge (such as Extended Basic) that you chose to "plug in" to your PC99 computer. If you have a joystick or two connected to your clone's game ports, the joysticks will behave like TI joysticks when running TI software. Parallel and serial ports on your clone are addressed as "PIO" and "RS232" just like the real 99/4A when running TI software.

NEXT PAGE

When you purchase PC99 you get the Tombstone City and Extended Basic "cartridges" as part of the initial purchase price. Cartridges and TI "disks" are files on your PC's PC99 subdirectory. The PC files that emulate TI "disks" do in fact behave exactly as if they were disks on a 99/4A system. When running PC99 you can OLD, SAVE, RUN, OPEN, CLOSE, and catalog the TI files that are on these PC99 "disks". Cartridges that run on PC99 are based on Gram disk files created on a real 99/4A system with a GramKracker, PGram, or similar Gram emulating device. If you don't have a gram device, the PC99 authors will sell you in PC99 format a legal copy of any TI cartridge ever produced. TI gets a royalty from such sales.

You always have access with PC99 to 3 DSSD "disks" and to any software on these disks. Converting actual physical TI disks (with all their Gram files, BASIC XBASIC EA3 and EA5 software, and their data files) to a PC99 "disk" file requires either cabling the IBM clone and 99/4A systems together via their serial ports or using the commercial software PC Transfer on a TI system with a double density floppy controller. The process is fairly straight foreward for converting module gram files, and a bit more complicated for other types of TI software. But it can be done with a little time and careful reading of instructions. So far I have converted a complete Funnelweb system, and about a dozen modules. Once the conversion is done you can save your IBM "cartridge" and "TI disk" files to IBM floppies for archival purposes. The conversion only has to be done once. Utilities are provided to convert individual "TI" files or whole "TI" disks both ways, from the TI to the clone and from the clone to an actual 99/4A. I suspect PC99 owners will trade a lot of these TI "disk" and "cartridge" IBM compatible files.

Changing cartridges is easy! Just exit PC99 and from DOS run a little file called "LC" which means LoadCartridge. Enter the file name followed by the name of the cartridge, such as "LC extended basic". Then the next time you start PC99, Extended Basic will be item #2 in the menu that follows the color bar title screen. Of course you need to have the PC99 extended basic gram files on your clone's hard drive to do this. One thing that takes some getting used to about "LC" is that cartridge names are case sensitive. In the above example, entering "LC Extended Basic" would not work because of the capital E and B. Most of the time DOS is not case sensitive, and LC runs out of DOS so you might think that case isn't important. It is!

There are some things you don't get with PC99. Because it only emulates official TI peripherals you only have three floppy drives available and these are only single density. PC99 does not emulate the double density 4 drive capacity of CorComp or other third party floppy controllers. 80 column software for the 99/4A and Geneve specific software will not run with PC99. You don't get to use ramdisks (you really don't need them), clock cards, or gram emulators with PC99. The current release of PC99 has no speech synthesis and only has one sound channel, because a typical clone has only one sound channel to its PC speaker. The next release is expected to support a PC sound card such as a "sound blaster". This will allow emulation of TI speech synthesis and full "3 channel plus noise generator" 99/4A sound.

Some additional observations about PC99, not necessarily in any order:

Speed- PC99 is either slow or very slow, depending on the speed of the PCs microprocessor. Typing in BASIC listings, or responding to INPUT statements from within a running BASIC program can be done at normal typing speed. Almost everything else is slow, particularly sprites and other graphic intensive applications. I am told that the Funnelweb text editor works at near normal speed on a 486/DX2 66 machine. It should run even faster with a Pentium processor. On my 386/DX 40, the machine I am using to write this article Funnelweb's text editor is SLOW, too slow to really be practical. I have to wait after each typed letter for the letter to appear on screen before typing the next letter. There seem to be no speed problems printing through the Funnelweb formatter using PC99. I have used PC99 on a 386SX 20 machine at work with acceptable but slow results. This is just about the slowest microprocessor in the 386 line.

Games- I have been able to achieve some fantastic scores on TI game cartridges, fantastic for me anyway. These cartridges run slow enough that I can anticipate what is going to happen. Keyboard response or joystick response (or maybe just my response time) from PC99 seems more responsive than on an actual 99/4A. I now have no trouble eating all the dots in car wars and advancing outside the village to kill all the bad guys in tombstone city, things I can't do on a real 99/4A.

Mathematical Accuracy- We pride ourselves on the 10 digit display and 13-14 digit accuracy of mathematical calculations done on a 99/4A. This is better than what you get using the various BASICs that run on IBM clones. On a 99/4A 1/3 x 3 = 1, whereas on many PCs it will equal 0.9999999 beacuse of rounding errors. PC99 is every bit as mathematically accurate as a 99/4A. This means that a 386 or 486 CPU on an IBM machine is NOT less accurate than a 9900 CPU on a 99/4A. The apparent lack of mathematical accuracy in PC BASICs is because of the way these BASICs are originally written in the PC's assembly language, not because of any deficiencies in 386 or 486 CPUs.

The 360K Double Density Disk Problem- Transferring disks and files from a 99/4A to PC99 requires either cabling the computers together or using PC Transfer. PC Transfer allows you to put an IBM formatted 360K 5.25 inch double density disk into a double sided double density drive of a 99/4A

NEXT PAGE

system that has a double density controller. You can then, using only the 99/4A computer, transfer files from 99/4A formatted disks to 360K IBM formatted disks. The problem is that most modern IBM clones with 386 or fancier processors have only "high density" floppy drives. These drives can be used to read low capacity "double density" floppies but cannot be reliably used to format a "double density" floppy to only 360K. The DOS instruction book says this can be done, but it really can't. You either have to have access to an older IBM clone (such as my Tandy 100HX) with a "double density" 5.25 inch floppy drive or you need to puchase preformatted 360K "double density" 5.25 inch floppies. For most people it is probably most convenient to purchase preformatted 360K disks. They are available at Radio Shack and some discount department stores.

The PC99 debugger- This is a superior assembly language tool that lets you set break points, move in single steps through software, etc. Since the PC99 debugger runs outside of the 99/4A environment, it allows access and analysis to ALL memory areas. No debugging tool running on a 99/4A can do this.

In conclusion: if you have a modern IBM compatible computer and you want to run all your favorite 99/4A software off your new machine, than PC99 is the only way to go. It seems to work perfectly, although PC99 is sometimes a bit slow. Cost for the current release is $120 from CaDD Electronics, 81 Prescott Rd., Raymond NH 03077. 603-895-0119

****DONE****

FIRESIDE CHAT #9

by Jim Krych

(BB&P editor's note: The Jim tells me that this article is in part a response to Tony McGovern's "Letter From Australia - No.6" originally published in the December 1993 issue of BB&P.)

My friends,

This promises to be the longest of my Fireside Chats to you about memory systems for the TI 99/4A Home Computer. This Fireside Chat is directed to those in the community who have only heard about the AMS or are still wondering what it is. This is also directed to those users/programmers who may be so used to the old ways that what we, the AEMS Project Team, have done seems to be totally wrong, with the old thinking. It is the most sincerest hope that this 'Chat shed a new light on a new way of doing work with the Texas Instruments 99/4A Home Computer. That I may not convinvce all readers is unimportant, all I ask is for one to read this and THEN base their judgement upon the memory system.

The Asgard Expanded Memory System(AEMS) Project was started in September of 1990. Our goal was to produce a memory system that would isolate the programmer from having to do the manual paging of memory. It needs to be pointed out that ALL memory cards for the 99/4A had to be programmed manually. This, we believed then as we still do, prevented any major work done on any expansion card. This was our goal: "Ease of Use Expanded Memory."

Not that it would be a ramdisk. Also, the system had to be efficient and fast, even for a 4A. This requirement precluded using any of the ramdisks as a model to work with. This also precluded any design to a ramdisk that allowed it's memory as CPU program space, RAMBO.

Our only model, for the 99/4A world, would be the TI 99/8 Computer, which was never released. But we also had the advantage of looking at how other memory systems for other computers worked. What we have come up with then is both a mix of what the 99/8 did as well as what is being done now in the computer world.

Ours is not the only machine that has faced the problem of accessing greater memory than the computer can handle. This problem of "fitting a mountain into a molehill" has been around the computer field almost since it's conception. Our problem is not unique, rather only the machine we implemented the memory system on was unique.

But this is not to say that others have tried. Indeed, from our rather cloudy history of Texas Instruments, comes word that even TI had planned for memory cards for the 99/4A. But we need to remember one very important fact, one that seems to be overlooked everytime one mentions that TI did design a 128k card, the 99/8 Home Computer had a built in memory mapping system.

The TI 99/8 utilized a memory mapper INTERNALLY, that is the memory system design was on the 99/8's motherboard. The reason that the 128k cards were never released for the 4A would be to not compete against the 99/8. In other words, TI did not wish to shoot itself in the foot, since the 99/8 was "forthcoming."

But the 99/8 was never released. And Texas Instruments abandoned the home computer market.

But this is not a criticism of that decision. This is an analysis of why did TI design the 99/8, with it's memory system, the way it did.

The 99/8 memory system used a memory mapper. It's registers lie in the >8000 area, know to us by the TI manuals as "reserved for future use." The 99/8 memory mapper took the four most significant bits of the address bus to select one of sixteen registers. Each register corresponded to a 4k page.

This system is efficient as well as fast, ignore for the moment that the 99/8 used a faster cpu-a modified 9995, at 10.7 MHz speed. Using a memory mapper allows for pointers to extra memory. Rather than physically banking memory in and out, memory mapping points to a new section, ie. a page, of memory. The result is an incredibly fast means of accessing greater amounts of memory.

The 99/8 used extended operations, XOPS, for its memory as well. But here lies the most important fact, the 99/8 used neither any Device Service Routines(DSR's) nor did any Communications Register Unit(CRU) for paging.

Lets explain here just why DSR's exist for us in the 4A world. DSR's are nothing mysterious. They are a cheap means of peripheral expansion to the 99/4A Home Computer. The other computer world has DSR's too, but they are called device drivers. Where our device drivers are accessed only once, theirs are in a continuos block of memory and can be in memory all at once.

But our computer also does something for DSR's as well. That is called polling. What does that mean? When the 99/4 was designed, the architects needed a way for future expansion to be possible, otherwise the computer would be rather limited. This computer would have a "peripheral bus" that would allow up to sixteen(16) different devices to be attached to this bus. If one doesn't count memory mapped peripherals, ie. speech, this amount of expansion is impressive, even for today.

When someone added a new card to this bus, with it's own DSR, it would complement the 99/4A's operating system-that is our device independent os that makes the 4A special. When one wished to access the new card they would type in the new call. if in BASIC/EXTENDED BASIC, or any other system, and the computer would start polling through it's peripheral bus.

What would it do? It would start at the beginng address. It would would look for a header code, something that would tell the computer "I'm here!" This header code is >AA. If it did not find a code, it would go on to the next address. Upon finding a header code, it would then do a comparison to the command typed to what was in the card's DSR. If the command(s) matched, the command would be carried out. If the system looked through it's bus and didn't find any matching code, an error would be produced, "I'm not here!"

This is how our computer works for peripheral expansion. The beauty of the system is it's cheapness. The disadvantage is that ANY call will make the computer poll through until the call is completed or an error given. This means that if for example, a card lies very near the end of the bus, the computer would search through the following DSR's, assuming a minimum TI expansion system:

Disk system
RS232 card(s)
Now say a user has the following, according to bus order:
Ramdisk
Disk system
RS232
HFDC Card(hard drive)
New device

Each card has it's own DSR. So any call to the "new device" would mean going through the above order. Starting with the Ramdisk and finally ending with the "New device." Notice that the Ramdisk is before the disk system, because of the headers used in the Ramdsik DSR, disk system access WILL SLOW DOWN.

A pattern is here, a very important one in the least! These are all peripherals, that is devices that are inherently much slower than the actual CPU of the computer system. Even a Ramdisk is nothing more than a solid state floppy drive. Faster than a floppy, oh yes, but no where near as fast as the computer itself.

Nowhere in the list was memory mentioned. Memory for a computer is a special type of peripheral. To achieve an efficient memory system, one would NOT burden the memory with the ineffieciency of a device. The 32k card does not have a DSR.

A DSR is important, make no bones about that. But for memory use, a DSR is in the way, rather than a help.

The AMS, Asgard Memory System, does not use a DSR. To some, that is an uforgivable deficiency. But the AMS family is not at all like other memory cards. Indeed, no other memory card is "equal" to the AMS. AMS is different. The only other memory system for the 4A that is similar is the Geneve, which uses a memory mapper as well.

The AMS design is fairly simple and inexpensive. As part of it's design, it acts as an exact copy of the original 32k card! This is the "PASS" mode of the system. It is true that the AMS uses a CRU location, the >1E00 space, BUT WE ONLY USE THAT TO TURN THE CARD FROM "PASS" TO "MAP" MODE! This is an important note to remember. "MAP" mode is where expanded memory is handled. The memory mapper we uses has been around since 1983! We are able to have up to 4096(12 bits) total pages, each being 4k in length. That is a total amount of 16384k memory. The AMS family uses the 8bit data bus of the 4A for a total of 256 4K pages, 1024k. Converting the bus would allow for full memory use.

Which brings about the page size. AMS uses a 4k page. To some, this is unexcusable. To which it must be said, point to another page. But manually programming the AMS is not what we have dedicated our resources to, rather to enable a stable foundation for a memory system.

NEXT PAGE

The system software for the AMS is truly unique. NOT ONE PACKAGE FOR THE 99/4A CAN CLAIM BEING AN EQUAL TO IT! The software team came up with a method that allows for greater than 32k programs to be designed and run, all without ever using a manual process.

This is something of extreme importance. The arguments for a DSR give examples of callable routines to enable expanded memory. BUT THIS STILL MEANS THAT THE PROGRAMMER MUST PHYSICALLY PAGE THE EXPANDED MEMORY, ALTHOUGH HE OR SHE MAY BE "ISOLATED" FROM THE LOW LEVEL ROUTINES.

Rather than explain totally the process, I will describe what the software team has done.

The new loader allows for either E/A Option 5 files or our own files to be loaded and kept "resident." Some may argue, that is EXACTLY WHAT A RAMDISK IS, which is an incorrect comparrison. Programs loaded as resident ARE IN MEMORY, once selected they run. A ramdisk would first load the program from itself then into the 32k memory. Plus, the major feature, is that the loader allows for far greater size programs to be run, one need not keep them, as resident. A game, TINOPOLY, is 72k in length! It is run through our loader.

A Macro Assembler. This is a total uprgade to the RAG Macro Assembler. To run this, one MUST HAVE an AMS. Some have said that a macro assembler is a gross overkill for the 32k 99/4A. But this new assembler is not for 32k, rather it is part of the new memory and part of our "Ease of Use" concept. Although the files it assembles can be for a regular 32k system, not just for AMS.

The Object Linker. This program, more so than anything else, is the embodiment of "Ease of Use." A programmer links his modules togetner-no manual paging, for a much larger program. The modules can be a minimum of 4k! A MINIMUM! They can go all the way up to 24k in length. For those who complain about "only 4k pages" that argument is moot.

Neither is the Linker a collection of linkable routines, some have erronously stated. The Linker not only proves "Ease of Use" but destroys FOREVER the myth that a DSR is needed for expanded memory use. Nowhere does the programmer call routines to handle the expanded memory paging, nowhere!

With this in mind, the next step would logically be the creation of routines to handle large data buffers and such. Those exist! Also, of far greater importance, is the availablity of the 32bit memory allocation routines. With these, a programmer can create large buffers, page wise, but manipulate them TO THE BYTE! To those who have cried for such tools, here they are!

I would like to point out here that none of which I have described is "being written as we speak" but rather, THEY EXIST NOW AND HAVE BEEN AVAILABLE FOR NEARLY SIX MONTHS!

This community has been plagued for far too long with promises. The AMS is real. This software is real. The software team has even modified c99 for use with the Linker! TINOPOLY, the 72k game, is a c99 and Assembly mix!

AMS is not equal to other cards, nor can any memory card claim to be of comparrison to AMS. AMS is at once an extension of the 99/4A and a fulfilment of what would have been. There is nothing similar to AMS.

So what has been done? Nothing! That is if you believe in the other cards. Something totally new! That is if you see what has been really accomplished. A totally new memory system! Not just a card! Not just a collection of callable routines! BUT A MEMORY SYSTEM!

Thank you! Take care and God Bless!

DONE

****
* BITS, BYTES & PIXELS *
* Published by Lima OH *
* 99/4A User Group *
*
* Material contained herein *
* may be copied by any user *
* group as long as credit *
* is given. DV80 files of *
* most articles in BB&P can *
* be obtained by sending a *
* disk and return postage. *
*
* ADDRESS- P.O. Box 647 *
* Venedocia Ohio *
* 45894 *
* Published monthly except *
* July and August *
* ---------- *
* GROUP OFFICERS *
* President-Susan Cummings *
* 419-295-4239 *
* Vice Pres-Peter Harklau *
* 419-234-8392 *
* Treasurer-Leonard Cummings *
* 419-738-3770 *
* Newsletter editor and *
* Librarian-Charles Good *
* 419-667-3131 *
****

FUNNELWEB "EVRAM" PROGRAMMER'S EDITORS and ASCII-HEX PROGRAM CONVERSION UTILITY

new software from Tony McGovern
described by Charles Good
Lima Ohio User Group

Tony McGovern has released "the first of several new versions of the Funnelweb v5.0 editor". This 80 column programmer's editor uses a fully expanded 9938/9958 video processor with 192K ram to permit a 64K text buffer. This a enormous by any TI standard, even larger than the text buffer of MYWORD running on a Geneve. In ShowDirectory you get a display of % free and number of text lines free. This new editor will handle over 3500 text lines. For technical reasons word wrap is permanently disabled. According to Tony word wrap would be too slow for most typists.

Although I don't write large blocks of assembly code, which is what this editor is designed to handle, I have found some uses for the extra text buffer. 1- Text files (such as CYC files) ported over via PC TRANSFER from an MS-DOS computer are often too large for the Funnelweb text editor. I can load them into the 64K program editor and then break them down conveniently into TI Writer sized segments. 2- When I create disk listings of Lima Library software I do this by printing DSKU disk reports to a disk file. The resulting DV80 disk file is often too large to fit into a normal text editor for editing, but easily fits into the new Funnelweb 64K editor.

Tony McGovern has also written a utility which converts PROGRAM files into a text file of the ASCII equivalent. The result is similar to the ASCII display of a sector editor. These ASCII text files can then be transferred to a PC disk using PC TRANSFER and uploaded to a BBS. Recipients of these ASCII text files can convert them back to PROGRAM files with the same utility. The ASCII-HEX conversion utility is an EA5 program. Run it, then type in the name of a file. If the file is a PROGRAM file you will get ASCII DV80 output. If the file is an ASCII file you will get output as a PROGRAM file. This is useful if your TI is not connected to the internet, but you know someone with a PC that is connected to the internet. You can then send and recieve TI PROGRAM files using MS-DOS computers, and convert them to/from TI disks using PC TRANSFER.

The 64K v5 editor is sharware. The ASCII-HEX conversion utility is public domain. They have been uploaded to GEnie and are also available to anyone who sends $1 to the Lima User Group, P.O. Box 647, Venedocia OH 45894.

I also have from Tony McGovern a preliminary version of the 80 column Program Editor that has TWO 64K buffers in which you can put text for editing without word wrap. What used to be the V(iew) buffer has been made into a second edit buffer. This means two gigantic text edit buffers simultaneously in memory. You can put different text into memory in each of the two text buffers for editing and you can quickly switch from one buffer to the other. You can also cut and paste between the two buffers. FANTASTIC! Tony says "Keep this one for yourself- it is not in shape yet- it is not a secret either."

DONE

MORE NEVER RELEASED OFFICIAL TI SOFTWARE

by Charles Good

Programs under development by TI and others for the 99/4A but never officially released keep appearing out of the woodwork.

SAGUARO CITY. Copyright Texas Instruments 1981. This looks like it is the original version of what TI officially released under the name "Tombstone City". We have it as an EA5 loadable program file on disk B57B. There are two graphic differences between this and Tombstone City. 1- The title shows as "Saguaro City". 2- There are Saguaro cacti constantly on display in the upper left and right of the screen, but dead bad guys (morgs) are displayed as tombstones in the desert outside of town. In the officially released game (Tombstone City) the dead morgs all look like cacti and there are no tombstones. It seems to me that TI got its names mixed up. The title "Tombstone City" should go with the one never released) that shows the tombstones, and "Saguaro City" should have been the title to the (officially released) version that shows all the cacti.

FANTASY. I had gram files of this game sitting around my software collection for some time, but I couldn't get them to work. Recently I carefully read my Gram Kracker and P-Gram documentation and began playing around with the headers of my FANTASY files. As a result of these efforts I now have a partially working version of FANTASY.

There is nothing in the printed literature concerning this game (nothing in the CYC, no advertisments in any 1983/84 computer magazines) and no mention of TI or a copyright on the game's title screen. There is little doubt however that this was under development by TI prior to abandoning the Home Computer in late 1983. I think this game was on TI's gram emulator, used by the company to demo 99/4A cartridge software at public exhibitions in 1983. Examining with a sector editor the Fantasy gram file that loads into RAM at <6000, I find the text "SOLID STATE CARTRIDGE" and "(c) 1983 TI". These text strings do not appear anywhere in the game, but suggest ownership by TI. "Solid State Cartridge" is an official TI service mark (trade mark) that can not legally be used by others.

NEXT PAGE

When the game starts you hear a female voice say "Help! Help!". Apparently it is your job to rescue her. You are given your choice of 1 or 2 players and then 1- Ship at sea 2-Ship's deck 3-Jungle 4-Jungle maze. These are apparently game lavels. Successful completion of one level automatically advances you to the next. The game uses joysticks or the split keyboard for movement and firing. Continuously, the girl says "Help!" or "Help me!".

1-Ship at sea. The girl is on a pirate ship. The hero is in a balloon which has to be guided to the ship's deck. The ship's cannons try to shoot down the balloon.

2-Ship's deck. Our hero has to fight off the ship's crew and get to the girl who is in a cabin in the center of the ship. I have never been able to do this successfully.

3-Jungle. Our hero is in a balloon floating over the jungle. Flying above the jungle are two large ugly birds which try to puncture the balloon. Our hero has to avoid the birds.

4-Jungle maze. This is a multi level "Donkey Kong" like maze. Our hero is at the bottom of the maze and has to climb to the top where the "Help help" girl can be seen. I can't get this level to work at all. The computer seems to be locked up.

Anyone who wants partially working gram files of this game can have them be sending me a disk and paid return mailer.

DONE

4A HINTS AND TIPS
By: Andy Frueh, Lima UG

WOW! Thanks a lot to the people that have sent me their tips! This is my kind of support. Here's to the next few installments! Just remember that your tip doesn't have to be a programming tip or utility. It can be a game strategy, a new idea for old equipment, or any little hint or tidbit will get published if it seems to be a new, innovative idea, or just plain nifty!

This edition, I feature tips sent in by Gene Bohot. He sent me a disk with several tips. He tells me he has a several disks full of such information. Let's see some more come in, folks. Make this YOUR column.

This little graphics program puts a helicopter against a night blue field with moving stars. It actually feels like you are watching a flying helicopter. Maybe someone can turn this into a game program.

```
10 CALL CLEAR :: CALL SCREEN(5)
20 CALL CHAR(33,"0000000000030000")
30 CALL CHAR(64,"000000000FF0000")
40 CALL CHAR(35,"000000000FF40E0")
50 CALL CHAR(94,"0000000000000008")
60 CALL CHAR(37,"0709113F3F1FB2FF")
70 CALL CHAR(38,"FC0E0F0F0FFE10FF")
80 CALL CHAR(42,"0000FFFF000000C0")
90 CALL CHAR(36,"1C3CFCFC00000000")
100 CALL CHAR(63,"0000000000000000")
110 CALL VCHAR(12,16,38)
120 CALL VCHAR(12,15,37)
130 CALL VCHAR(12,17,42)
140 CALL VCHAR(12,18,36)
150 CALL VCHAR(11,18,94)
152 FOR I=1 TO 15
154 R=INT(RND*180)+10 :: IF (R>70)*(R<110)THEN 154
156 C=INT(RND*180)+10 ::
   CALL SPRITE(#I,46,15,R,C,0,+15):: NEXT I
160 CALL VCHAR(11,14,33)
170 CALL VCHAR(11,15,64)
180 CALL VCHAR(11,16,35)
190 CALL VCHAR(11,17,64)
200 CALL VCHAR(11.14.63)
210 CALL VCHAR(11,15,63)
220 CALL VCHAR(11,16,63)
230 CALL VCHAR(11,17,63)
240 CALL SOUND(135,-4,0)
250 GOTO 160
```

As most of you know, I like programming music. I belive that the TI sound chip is probably one of the most sophisiticated in that several special effects can be easily programmed. For example, here are two examples of special effects using dulcimer and tremolo routines. While the music is simple, you get the point.

```
1 ! DULCIMER
2 CALL CLEAR :: DIM S(26):: F=262 :: FOR N=1 TO 25 ::
  S(N)=INT(F*1.059463094^(N-1)):: NEXT N :: READ N ::
  C=S(N):: D=S(N)
3 RESTORE 7 :: FOR J=1 TO 63 :: GOSUB 5 :: NEXT J
4 U=U+2 :: CALL SOUND(-200,S(N),U,C,U,D,U)::
  IF U>27 THEN U=0 :: GOTO 3 ELSE 4
5 READ N :: CALL SOUND(-500,S(N),0)::
  CALL SOUND(-500,S(N),0,C,9)::
  CALL SOUND(-500,S(N),0,C,9,D,19):: D=C
6 C=S(N):: RETURN
7 DATA 5,6,8,8,10,13,5,5,6,5,3,3,5,3,1,1
8 DATA 5,6,8,8,10,13,5,5,6,5,3,3,5,3,1,1
9 DATA 8,13,17,17,17,15,13,13,8,8,10,10,13,10,8,8
10 DATA 1,1,1,3,5,5,8,5,3,3,5,3,1,1,1

100 ! DEMO OF TREMOLO
110 FOR J=1 TO 60 STEP 2
120 READ A,B
130 FOR L=1 TO A
140 CALL SOUND(-1000,B,0)
150 CALL SOUND(-1000,B*1.02,0)
160 NEXT L
170 NEXT J
180 CALL SOUND(-1,30000,30)
190 DATA 2,330,2,294,4,330,4,294,4,330,4,294,4,262,8,220
200 DATA 2,330,2,294,6,330,2,294,4,330,4,262,12,247
210 DATA 2,294,2,262,4,294,4,262,4,294,2,330,2,294,4,262,8,220
220 DATA 4,262,4,262,4,220,4,262,4,247,16,220
```

NEXT PAGE

I also like special screen effects. These can usually be programmed painlessly before or after the rest of a program is put together. The following are examples. The first program places a box around a certain area of the screen. The next example is a neat idea for an attention mark, ect.

```
10 CALL CLEAR
100 R=1 :: C=3 :: NC=27 :: NR=19
105 GOSUB 10000
110 DISPLAY AT(8,8)SIZE(13):"-BOX- PROGRAM"
120 DISPLAY AT(12,8)SIZE(13):"PRESS ANY KEY"
130 CALL KEY(0,K,S):: IF S=0 THEN 130
140 END
10000 CALL CHAR(95,"FF00")::
      CALL CHAR(125,"00000000000000FF")::
      CALL CHAR(93,"8080808080808080")::
      CALL CHAR(91,"0101010101010101")
10010 CALL HCHAR(R,C,95,NC):: CALL VCHAR(R,C,91,NR)::
      CALL HCHAR(R+(NR-1),C+1,125,NC)::
      CALL VCHAR(R,C+NC,93,NR)
10020 RETURN
```

```
100 CALL CLEAR
110 DISPLAY AT(10,7):"A T T E N T I O N"
115 CALL SOUND(20,800,0)
120 FOR X=1 TO 90 :: NEXT X
130 DISPLAY AT(10,7):"    attention     "
135 CALL SOUND(20,400,0)
140 FOR X=1 TO 90 :: NEXT X
150 GOTO 110
```

Most people do know that I think the TI is suitable for games. Well, I don't use it for games quite as much anymore, but Gene sent me this game. It is unique in that it is both fun and educational. Of course, after a while this type of game will get boring, but just the idea itself is great.

```
10 RANDOMIZE :: CALL CLEAR ::
   ! BALLISTICS SIMULATION WITH SPRITES
20 DISPLAY AT(4,2):"SHOW PATH (Y/N) Y" ::
   ACCEPT AT(4,18)SIZE(-1):@$
30 CALL CLEAR :: L=45 :: VV=55
40 FOR N=1 TO 14 :: CALL COLOR(N,2,15):: NEXT N ::
   RR=INT(20*RND)+10
50 FOR A=1 TO 28 :: CALL SPRITE(#A,42,2,256,1):: NEXT A ::
   CALL SCREEN(6)
60 CALL VCHAR(1,3,124,24):: CALL HCHAR(24,1,95,32)::
   CALL HCHAR(24,RR+2,31)
70 DISPLAY AT(4,2):"TARGET=";RR ::
80 DISPLAY AT(1,2):"ANGLE (11-86)=)";L ::
   ACCEPT AT(1,18)SIZE(-2):L
90 DISPLAY AT(2,2):"SPEED (10-70)=)";VV ::
   ACCEPT AT(2,18)SIZE(-3):VV
100 IF VV>70 THEN 90 ELSE V=VV/10
110 A=PI*L/180 :: R=V*V*SIN(2*A):: COSA=COS(A):: TANA=TAN(A)
120 DISPLAY AT(3,2):"RANGE=";R :: X=0 :: CL=(ABS(RR-R)<=1)
130 FOR XX=0 TO R-.1 STEP R/28 :: X=X+1 ::
    Y=(-1/(2*V*V*COSA^2))*(XX^2)+XX*TANA
140 IF @$="Y" THEN J=X ELSE J=1
150 IF Y<=24 AND Y>0 AND XX<=30 THEN
    CALL LOCATE(#J,192-8*Y,8*XX+12)
160 NEXT XX :: CALL SOUND(-100,220,0,-7,0)::
    IF CL THEN CALL SCREEN(7):: GOTO 30
165 CALL KEY(0,K,S):: IF S=0 THEN 165
170 CALL CLEAR :: GOTO 50
```

I have seen many ways of changing the colors in immediate mode with the XB cartridge and 32K memory, but most are erased when either an error, or a NEW is encountered. This routine stays unless you QUIT, but it intereferes with most assembly and CALL LOAD routines. Funnelweb users beware!

```
200 CALL CLEAR
210 INPUT "Screen Color (1-16)? ":A
220 INPUT "Text Color   (1-16)? ":B
230 C=(B-1)*16+(A-1)
240 CALL INIT :: CALL
    LOAD(9984,C,C,C,C,C,C,C,C,2,0,7,15+A,4,32,32)
250 CALL LOAD(9999,48,2,0,8,0,2,1,39,0,2,2,0,8,4,32,32,
    36,2,0,8,8,4)
260 CALL LOAD(10021,32,32,36,2,0,8,16,4,32,32,36,2,0,8,
    24,4,32,32,36,4,91)
270 CALL LOAD(-31804,39,8)::
    CALL LOAD(-31952,255,231,255,231)
```

To break protection on programs, enter the following.

```
CALL INIT :: CALL LOAD(-32187,0)
```

I know that CALL LOAD(-31931,0) does the same thing, but this one also appears to work. This is one of the little secrets that TI wouldn't tell us. Now, there is a way to back out of the PROTECT clause protection scheme. The XB manual says there is no way, and there was at one time a commercial program that did this (it sold for about $10!). Now, you can do it for free.

Well, I feel that it's time to close out this set of tips. Gene Bohot's tips will continue at the next installment. Remember to send in your tips. Surely you have some useful knowledge that can be shared.

****DONE****

CORRECT DATES FOR NEXT LIMA CONFERENCE

The correct dates for the 1994 Lima all TI/Geneve conference are FRIDAY MAY 13 and SATURDAY MAY 14. It has been stated in error elsewhere that the dates were May 14/15. We apologize for any confusion.