THE TACOMA INFORMER

8905

Published by The TACOMA 99ers USERS GROUP
P.O. BOX 42383 -- TACOMA, WA 98442

Volume 6 | MAY 1989 | No.5

GET THE SHOW ON THE ROAD

by JOE NOLLAN

I will be pounding out some articles in the future, but for the present let me say that the Tacoma 99'ers are alive. The drives and controller card for our second expansion system are on the way. This second system will be upgraded to the 9640 in the future. I will be continuing the series of talks and demo's of the commands and statements to hone our programming skills. The subject matter of my demo is open to suggestions, so if you have any burning questions, let me know. I have not done much with my computer lately. I have revised my SYSTEM disk to reflect the new year (1989) in the title appearance, and will get a copy of it into the library. I might add that I did receive $10 from a group in England, thanking me for my efforts.

Since I don't have any projects on the burner, I have the time to work on Library updates/clean outs. If you have a problem with a program, let me know. I don't run all the programs, very few in fact, so I'm not familiar with everything in the Library. This is also a good time to get my help on a program that you may be working on.

I also noticed a writeup about the Computer Room in the Park's flyer so we will have to put forth the effort to have it staffed.

For those who do elect to call me, you will have to work through the answering machine. It has cut out the many and varied salespeople who invade my privacy (always at dinner time). I don't like missing calls from people that I know, and I find it interesting that not one salesman has ever left a message.

At the meetings I have been doing a series of DEMO's of different programming statements. I plan to continue to do this as long as there are questions and interest. I am willing to use a particular problem as an example. Doing this will solve your problem as well as being a great teaching tool.

It's great to see the group back up to par. The cold weather put a big damper on meeting attendence. Our two systems will be put to good use. I plan to get an equipment check done on the other computer/TV setups for the Computer Room.

So we'll see what is new coming up and until then, have a great Spring.

WHY PROGRAM?
by JOE NOLLAN

One of the nicest features of a computer is it's programability. A program can be written to do exactly what you want. This of course will require some programming skill, but it is also the best way to get that skill. I own a rental property and wrote a simple program to calculate the depreciation. It does exactly that, no more, no less. In our work or home we will often come across problems that could be solved on a computer if only we had a program. The point that I am making is that with a little reading and a lot of patience you will get the program that you want. You will also get the programming skill to make the next problem solver easier to write. I recommend starting with a specific problem first. Defining the problem is the most difficult part. By starting with a specific problem you need not concern yourself with additional frills needed to please others. These can be added later by them.

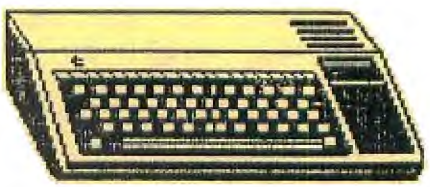
As an example, my wife and I run a Snack Shack for a Friday evening Fun night and are responsible for stock and accounting. I started with a simple list done on TI Writer. I could print the list showing the previous inventory and have a space to pencil in the new values.

I wanted some calculations done, totals of the cost amounts and sales. The list is only 25 items and it is possible to tally it on a calculator, but it takes time. I chose to use the computer, so I wrote a program that read the TI Writer file and did the calculations for me. My program would also make a new file with a list of stock on hand for the next week.

The program is simple, only 5 sectors, and does just what I want it to do. For me it was only about an hour of programming and it will save me time in the future as well as calculator batteries.

As I said before, we all have simple tasks for our computers to do. We do our part, write the program, and the computer will do it's part. The key is to sit down and do it.

I want to encourage each of you to put your machine to work solving your own particular problem. Remember, that I am always available to help in any way.



```
**********************************
*                                *
*   TACOMA 99ERS OFFICERS 1989   *
*                                *
*                                *
* President.....Frank Ashburn   *
*                      582-8886  *
*                                *
* V.President.....Tom King      *
*                      535-2563  *
*                                *
* Treasurer.....Ben Anstey      *
*                      535-1444  *
*                                *
* Secretary.....Joe Nolan       *
*                      572-4680  *
*                                *
* Librarian.....Bob Haun        *
*                      564-3938  *
*                                *
* Editor.....Jean Schmidt       *
*                      735-9056  *
*                                *
**********************************
```

FOUR-A/TALK

Random ramblings
about things TI.

by Bill Gaskill

April 1989

MYARC'S HARD DISK SYSTEM

Because I am a died-in-the-wool 99/4A loyalist (though my wife has another name for it) I took the plunge in September 1988 and purchased the Myarc Hard and Floppy Disk Controller (HFDC) card. I also purchased a Seagate 125-1 20 megabyte hard drive and power supply from Myarc to go with it. Total price was $735. A lot of bucks for a 99/4A computer peripheral you say? Yes, but read on before you pass judgement.

If you are like me you want your computer to be productive right out of the box. None of this having to be an electronics engineer to assemble it first, for me. I want components that say NO ASSEMBLY REQUIRED. I am simply not a hardware hacker and I am more than willing to pay a few extra dollars for the convenience and time and frustration savings realized in buying "ready-to-run" components.

When I decided to investigate the HFDC I called Myarc at their Georgia office and talked to Jack Riley. I expressed my concerns about "turn-key" components to him and was assured that Myarc could provide me with a system that required a minimal amount of assembly. Thus convinced, I placed the order and anxiously awaited its arrival. About three weeks later I called back and inquired about the status of the order. I talked to Jack again and he informed me that I would receive the power supply from Myarc-Georgia and the controller card and hard disk from Myarc-New Jersey within the next 4-5 days. I didn't listen close enough to what he said. If I had, I might have asked how a "turn-key" system could be delivered from two different locations, especially when the power supply would come from one place and the hard disk that goes in the power supply would come from another shipping point.

True to Jack's prediction, the hard disk and controller card arrived four days later, the power supply a day after that. When I opened the box containing the power supply I discovered that it was an off-white, steel cased enclosure measuring about 6 inches wide by 18 inches deep and 4 inches high. A handsome piece of equipment that made a nice looking addition to my workstation. When I looked for the instructions on what to do with it there were none. None meaning NOT ONE! A slight trickling of panic seeped into my thoughts. I then opened the box containing the controller card and found a 3-ring binder with about 80 pages of some of the most confusing documentation I have ever read. But not one word of it told me what to do with the power supply nor how it was to be hooked up to the hard drive. So much for "turn-key" systems.

Well I rolled up my sleeves, dug in, and figured out a couple of things without too much hassle. First, there are two power connectors inside the power supply box (white plastic connectors on the end of red wires) and you can use either one, since the unit is built to handle two hard drives. Second, the cover plate I got that fills the blank

space for the second hard disk I didn't buy, can be installed without a degree in mechanical engineering. The electrical cord for the power supply only plugs in one way so I couldn't very well do too much damage there. On to the hard disk.

The Seagate ST125-1 that I ordered from Myarc was represented as a 28 millisecond access time hard disk in my conversation with Jack Riley. I find that difficult to believe since Computer Shopper ads put the ST125 at 40 milliseconds. The ST125 sells for around $269. A 28 millisecond Seagate goes for just under $500. You can draw your own conclusions. Anyway, the drive is a 3.5 inch masterpiece that bolts easily to the base of the power supply. The back of it has two edge connectors for two ribbon cables that I could not find in any of the boxes that were delivered to my house. So I set the entire unit aside and moved on to the controller card.

In an act totally out of character for me, I read the manual before proceeding with the installation of my HFDC. When I was sure that I got all that my tiny brain could get out of the documentation I called Myarc-Georgia for help. I just couldn't make enough sense out of what was being said to risk damaging my $735 investment. It turned out that the phone call was about the first really correct thing that I had done (Jack Riley to the rescue). I immediately asked about the missing cables and was told to unfold the two panels in the HFDC box that kept the 3-ring binder and card from moving around. Low and behold, two ribbon cables, one 20 pin and one 34 pin. Boy did I feel stupid. From that point on, I decided to invest in the cost of whatever number of telephone calls it took to get Jack to walk me through the set up of the whole system. That was the second correct thing I did.

The installation went fairly smooth once I had a guiding voice on the phone to keep me from making any mistakes. In fact, the entire set up is rather painless once you go through it about two times, which I did. I don't know what went wrong the first time, but things got so scrambled on the hard disk that nothing would work right. Jack Riley offered the opinion that the cause was not turning everything on and off simultaneously with a master switch. I'm not totally convinced of that though since I have been able to turn the hard disk on last and off first and have had no problems. But I suggest that you stay with what the manufacturer tells you to do, so that your warranty isn't jeopardized.

There are some things about the set up, particularly the formatting of the hard disk, that I don't think I could ever have figured out on my own. When the FORMAT command is accessed from the Myarc Disk Manager 5 you must respond to a series of questions like volume name, sectors per track, number of heads, cylinders, write pre-compensation, interlace and the like. It all sounded pretty foreign to me. But with my guide on the other end of the telephone line I tackled the job as any brave soldier would who was facing the enemy's ground troops from behind an M60 tank. No problem!

In looking back, I again realize the correctness of making the telephone call to Myarc. Because the MDM5 disk manager program has gone through several updates and the documentation has not been brought totally up to date with the program, the manual tells you things that are no longer correct or that are incomplete or are in some way different than what you are staring at on your monitor. Had I not had the benefit of Jack Riley's immediate tutledge over the phone I would probably still be wondering how to get the HFDC and hard drive working.

I must fault Myarc for that. There is no excuse for not providing the most correct and up to date information in the manual. The documentation appears to be about two years old as far as I can tell and it gives such erroneous information as "the cables (connecting the hard disk to the HFDC) can only be put on one way. Wrong! The red strip on the edge of both of my cables had to be at the bottom when installed on the HFDC and facing to the right when attached to the power supply. Physically, they can be installed on the edge connectors either way, but only one way is correct. The manual really needs a going over to bring it up to speed with the current requirements of the set up process.

Back to the formatting. John Kolean wrote two nice articles in the September '88 issue of MICROpendium on the HFDC and its use with a hard drive. Some of the information helped me but not enough of it to get the system up and running. Here is what I did with my hard drive formatting. Maybe it will help you.

VOLUME NAME: 1 (The Volume Name is just like the disk name that you give a floppy disk. WDS1. already appears on the screen. You merely type in a name after the WDS1. to name the ROOT directory of the hard drive. If you name it the number 1 (the actual number 1) you will be able to read and write to your hard drive from programs that will not support pathing, which is most of the programs I own. I guess few software authors ever believed that we would ever get out of the DSK arena. Too bad. More on this later.

SECTORS PER TRACK: 32
HEADS: 4
CYLINDERS: 615
RESERVED DIR/FILE SECTORS: 2048
WRITE PRECOMPENSATION: 0 (erase the default then enter zero).
REDUCED WRITE CURRENT: 0 (erase the default and enter zero).
SECTOR VERIFICATION RETRIES: 8
INTERLACE: 2
STEP RATE: 0

These settings may not be valid for your drive if you have a different hard disk or an earlier or later version of MDM5. My copy of MDM5 is V1.25. I noticed that in the MICROpendium articles mentioned above John Kolean has V1.23 and the prompts he lists in the article are different than the ones that appear on my FORMAT screen. Versions 1.26, 1.27 and V1.28 are pretty much the same as 1.25, especially as far as what you are prompted for when formatting the hard drive. When I last talked to Jack Riley (March 13th) V1.29 had just or was about to be released.

When you have completed entering the above values type in the word FORMAT to format the hard disk. While the process of formatting is taking place MDM5 will dislay a bunch of information on the screen to give you something to stare at while it takes the six minutes or so to get the job done. If you begin noticing more than 1 or 2 BAD SECTORS appearing during the format you might have a problem with some kind of electronic garbage hindering the process. This happened to me so I tried reformatting again as John Kolean suggested in his article, but the problem only worsened. The formatting really slowed down and it seemed as if every sixth sector or so showed up as being bad. I KNEW that couldn't be right so I finally powered down the entire system and let it sit for about 10 minutes to clear it of any garbage and to let it cool down. That seemed to work as I was able to reformat the disk successfully on the next attempt, with no bad sectors.

Once the hard disk is formatted you can leave it as it is or you can create subdirectories on it. Either way, the system is usable once the formatting is complete. With my system I chose to build subdirectories to take advantage of the DSK1 emulation available with the hard drive. I also put a copy of MDM5 on my disk for lightening quick access. The configuration listed below has worked flawlessly thus far. If you can't get yours to work feel free to try mine if you like. I give no guarantees though, as I know only enough about this piece of hardware at this point to be truly dangerous.

Subdirectories:

WDS1. (the root directory)

DSK.
TIMP. (this is a nested subdirectory of the DSK subdirectory. It contains the files from my Multiplan disk: MPBASE, MPCHAR, MPDATA, MPHLP, MPINTR, OVERLAY).

DSK1. (this subdirectory contains the MDM, MDM5 and MDMXB files from the Myarc Disk Manager).

UTIL. (UTIL has the MDM, MDM-BACKUP, MDM-DELETE, MDM5, MDM6, MDM7 MDM8, MDM9, MDMCBACKUP and MDMXB files in it).

The process of making the subdirectories and copying the files to the various areas is fare for another article, but I will say that using the MDM5 program was and is a most enjoyable experience. It is truly a neat piece of software that is pretty easy to understand as it is. Not much is needed in the way of docs. Wonder who wrote it?

I still have a long way to go before I completely understand the workings and capabilities of my new piece of hardware, but I am absolutely thrilled to have it as an addition to my workstation. I now have what I will boldly claim to be the largest 99/4A data base in existence, 6515 records (2448 sectors) in a single file. I certainly could never have done that on any other storage media available for the TI.

Greg Anderson, a friend of Terry Hoder has a system for sale. PE Box with 1 single density 1/2 Ht single Density Qume Drive, 32 K Memory, TI Disk Controller, Console, Joysticks, X-Basic, Tape Recorder, P-Code and Manual. Best Offer. (206) 759 9481 or (206) 533-3872 Add. 516 24th St. Hoquiam Wa 98550

Tech-Talk by Mike Schmidt (cont. from page 9)

64k of total address space (via memory paging in the MYARC or HORIZON ram cards), larger programs may occupy that memory and give our TI's a greater running capability. The IBM uses a segment register that is pre-decoded to page in banks of memory, which is essentially the same way the HRD or MYARC does it, so memory expansion is no problem. The safe area in the TI is the first ROM bank, which is the invaluable interrupt routine and powerup routines. the SUPERCART is the only save RAM alternative for a kernel or DOS, since it is battery backed and it remembers all the changes you have made to DOS. In the CRU, the only area you could use for your own bit-twiddling is the >400 to >1000 area, which is not decoded presently and could be wired to something (I will let you imagine that). It would not be a difficult task to interface an IBM card to the TI, provided you had the correct cross-wiring, and a ROM to control the new device. A few of us in the chicago users group will attempt this. The price of IBM cards is falling like a rock, and I don't see any interfacing pitfalls.

TechTalk
by Mike Maksimik

The following came from the Chicago U.G. "S.I.G." on DELPHI:

Some of you may have followed TI's developments in the time that the 99/4A was at it's childhood. All sorts of plans, marvels, new things for the home computer that "was ahead of it's time." There were several peripherals developed by TI but were only released in tiny quantities, mostly to the TI employees that got the pick of the crop. Some of these never made it to the production lines, but only a few prototypes survived.

The modem card, which essentially was a Novation Cat 300 baud modem, was placed on a peripheral card, and a DSR ROM was given it to control very low-level functions, such as modem-to-vdp RAM interrupt routine, powerup routine, etc. It would work with a command module, like TE II just as the disk manager module works with the low-level routines in the disk controller to perform the DOS functions. Only a very few of these survived. Another little known card was the IEEE 488 bus controller card. It contained the TMS9914 GPIB (general purpose interface bus) that allowed the lab and mechanical equipment that used GPIB to interface to the TI. One could access the GPIB like a file device. This same standard is found in unexpected places. Any of you have a commodore 64? The communications bus used to connect it's ring-style bus of peripherals is a modified GPIB, one of commodore's own design. The SCSI interface (small computer systess interface) is essentially a multi-GPIB, allowing very fast buffered serial transfer between storage devices. SCSI also has interrupt lines to alert the host that data is waiting to be read or written. The VCR controller, a $500.00 range peripheral, along with support software, was introduced as a means to combine video from a VCR and the video from a TI. The card would control playback, hold, framing, and other functions. Digital Research created a similar product to control videodiscs that attached to an apple or a commodore 64, although much later than TI's development. The debugger card, a little known device, was in existence when the 99/4A was born. In fact, it's design can be rooted to the support hardware in the 990 minicomputer series. Essentially, the TMS9900 is a minicosputer on a chip. The editor/assembler GROM was a virtual image of the DX10 assembler used on the 990 minicomupter. Some directives one would only find on a minicomputer exist in the editor/assembler package, but were dormant in the 99/4A. The debugger board was designed to bring the 99/4A closer to a minicomputer's environment. The DEBUG program, included with the editor/assembler package, has several features that cannot be used without this piece of hardware. In fact, the editor/assembler looks as if it was taken direct from a 990 itself. The only added features were the GROM utilities, sucha VMBW, DSRLNK, LOADER, etc. that didn't support the features that a 990 could handle. It's too bad that TI wishes to keep the plans for this card on ice, it would be a dream to program with. It allowed multiple breakpoints by using the XOP 3 opcode, which would allow you to step your program through and look for errors or miscalculations. Although we can do this through software, the debugger board used a hardware approach. The design of this board, and what it contained, are up for grabs. If anybody knows, i'd appreciate you sharing with the rest of us. Send me a letter.

Still another rare peripheral was the GROM library peripheral. It essentially was a super-widget that could access ALL of the GROM in the cartridges. This would be handy for TI BASIC, since TI BASIC searches external GROM for subprograms. TI extended BASIC does this too, but doesn't search DSR ROM when a program is running. Modules like TE II, personal record keeping, and extended BASIC could all be plugged in and the CALL routines could be accessible to BASIC. BASIC could use the commands it wished to whatever, and all you had to do is plug your favorite "flavor" modules into the library peripheral to get the necessary language expansion. Imagine a GROM cartridge giving advanced graphics to TI BASIC, another for print spooling, still another for expansion memory control. Others for high speed cassette routines, etc. so the language could expand by adding cartridges. It's the same technique used with the peripherals: the computer never becomes obsolete, because it automatically responds to any new device attached. This is true of the library peripheral. This is another device I would LOVE to see.

Some of us have the HEX-BUS controller. In the days of the 99/2, the CC40, and the 99/8, the hex-bus controller was introduced for the 99/4A to allow compatibility with these devices. Essentially, they were designed like the commodore 64's peripheral system, where a slow serial transfer was appropriate for the hex-bus devices, a disk drive wouldn't be feasible. So TI never considered the HEX-BUS disk drive. The Wafertape drive, the CAT modem, the RS232/parallel interface, and the 4-color printer, were all developed. All were battery operated and could fit in a briefcase, as did the CC40. For the 99/4A, it was an inexpensive seans to expand.The hex-bus controller was a small device containing a DSR ROM that controlled the I/O drivers which "spoke" to the hex-bus peripherals. Since the main use was for the CC40, it wasn't pushed for the 99/4A. The 99/8 could also rely on the PE BOX for it's devices. It had it's own special FLEX CABLE card, which used some special

control lines to expand it's own capabilities. Since the 99/8 used a TMS9995, the same as the GENEVE, it could use the extra 3 address lines in the PE BOX, giving a total address space of 2 to the 19th power, or 512 k of directly addressable memory. Since some of these banks were probably switched, the address space grew to a total of 4096 k, which is sufficient for MOST of my needs. The speed of this processor was greater, and it's throughput was even greater, but more on that later. Some other control lines were used, some to indicate a 9900 or a 9995 present in the system, some to allow multi-level interrupts, still others to initiate HOLD sequences, which are found on the mainframes, and large multi-user systems as a way to deal with wasteful processing, and interrupt idling.

TI had a HARD DISK controller in the plans, probably MYARC's, but the technical data I have is 1982. I own a rare card. Some of you may remember a company called A/D electronics, out of Sacramento, California. They produced a control card which allowed sampling of environmental data through an 8-bit analog-to-digital controller. This device allowed hookups of many items, such as temperature probes, light transducers, etc. and was mainly used as a scientific device. Some possible uses included home control, because it also contained a real-time battery backed clock. Plus, there were separate digital inputs and outputs, for switches and relays, respectively. My main use for the A/D card, FIRST ADE, is a mouse. The RADIO SHACK color mouse contains two potentiometers turned by a rolling motion of the mouse. The potentiometers, when interfaced with the ADC0809 chip, (two channels, x and y) gives me mouse control with TI ARTIST. I wrote the DSR myself, and have been using this device for about a year and a half. The MBP clock card is a similar device, although it does not contain a digital input or output array. The ADE card, however, could also switch external relays, or sample data on 16 lines (8 in, 8 out). If timing was correct, an 8-bit parallel interface was possible. I still use this card, and the clock is handy for keeping my p-system master disk up-to date.

The FORTi music card was a device which allowed one to produce sound on not one but 4 extra TMS9919 sound generators. By arranging the frequencies on the 12 music channels available, different waveforms were possible. Now, with the FORTi, sounds even a c-64 owner could envy were possible. And, there were 4 percussion channels independent of each other. I can imagine "AXEL-F" running on this card!! And of course, we all know of the more common peripherals, the triple tech, the disk controllers, the 32k cards, te rs232 cards. Even these make our computers sophisticated enough to meet TI's long dead expectations. I also own the p-code card, and another article is devoted to THAT!

I mentioned the TMS9995 earlier. Just what exactly is a pipeline microprocessor? Well, the 9995 is not only fast, but it has a distinct advantage over others in it's class, even the intel 80386. Those processors rely on expanded address lines and increased instructions to increase throughput. There was a deeper approach, one that TI envisioned in the 9995. A pipeline microprocessor is one that incorporates special hardware that allows it to have more than one part of the microprocessor running at the same time. These CONCURRENT functions provide that while one instruction is being decoded inside the chip, another is being fetched from memory. Still another is being executed after it has been decoded. At best, with top-down code, and very little jumps, the microprocessor can achieve a throughput 3 times, or more, depending on the level of pipelining, over a regular processor running at that speed. For example, if we put test code into a 9995 and a 9900 running at 12 MHZ, the worst case is that the two run even. But the 9995 can pipeline, and with the pre-fetch and post-store the 9995 can LOOK like it's running 16, 20, or even 24 MHZ. And with the reduced instruction set in the control ROM, the 9995 has a distinct advantage over an 80386, it's MUCH cheaper to produce. The control ROM is a hard-wired design, while the 80386 has to be programmed externally. It is an easy device to interface to a memory system, and with no-wait state static RAM, the memory-9995 combination (up to 4 megabytes) can be phenomenal.

Currently, I am working on a software project. It's a new DOS for the TI, somewhat reminiscent of COMMAND DOS that ryte data released some years ago. However, there is no image file required because the DOS I have resides in a E/A supercart, and the utilities that it needs are extracted from the E/A GROM--that way, I can restore the lower memory expansion to a defined state very quickly without reading from a disk drive. The DOS is completely self contained, and will provide a choice for you on the master title screen. I am a college student, doing projects to complete my final years of undergraduate study in computer science. This project was inspired by a need for a better operating environment for the TI as well as a need for me to see if it could be done. Well, I have succeeded! The DOS uses the DSRLNK utility to attach to the low level device drivers. It gives you the familiar A> DOS prompt, and will mimic DOS to a degree, but with one delightful exception--the DOS is being written by me, and I can have it do whatever I want it to! I will no longer be a slave to incomplete DOS commands or ambiguous and useless

syntax, often the product of overpaid software developers. The commands are clear and precise, and the DOS is very short, only about 5k at this writing. Since most of the DOS is already present in our machines, in places like the E/A GROM, the disk controller ROM, the RS232 interrupt routine--all of these put together with the right glue can make a great DOS, and all I did was to provide the necessary glue for the parts, and it works! It has a batch file load and execute, D/F 80 loader (compressed/uncompressed), program file loader, dos utilities (FORMAT, COPY, RENAME, DELETE, ASSIGN) and screen control commands (WAIT, BEEP, CLS, GOTOXY, PRINT, ECHO ON/OFF) and "smart" control keys, as well as a 255 character input queue for type-ahead. Many of the commands are internal, and they reside only in the supercart. Other commands can be created from object code, which you can create from any one of the compiling languages, or the assembler (i prefer the assembler) and by simply typing the name of the file at the command prompt, the file will be loaded and executed.

I hope to have some sort of language compiler for DOS, such as a basic/pascal compiler, to facilitate creation of programs and utilities. My plans include a file transfer utility (terminal emulator), windowing, an 80-column editor, and multiprogramming. If for no other reason, then to gain experience and to enjoy doing it on my $49.99 TI99/4A. Of course, I wouldn't dream of charging anyone for this DOS, and I've had some interesting suggestions for names. "F-DOS" by our own editor, BOB DEMETER, for FROGMAN-DOS, since my "other" hobby is SCUBA DIVING, "XIOS" for eXtended Input Output System, and whatever...I am using version 1.24, which is relatively complete. I would just like to add the bells and whistles, plus write a manual on it's use.

Now for some more TechTalk. If you are confused as to why computers like the c-64 and the apple all have DOS commands built in...well, the designers of those computers anticipated a disk system, and available to most users, so the operating system and BASIC language all had the DOS commands either in the disk unit itself, or in a disk BASIC which loaded in on powerup. Since TI did things a little differently, they preferred to make DOS a separate thing, with a disk manager module to handle disk tests and formatting. It seemed a little annoying that in order to rename a file from BASIC, you had to either load the program and save it under another name, or if it was a DATA file, you had to OPEN it and read all of the data, then re-save the data to disk under another OPENed file name. This could be terribly inconvenient to users, but consider what the others have...the c-64 must send all of it's DOS commands through a command channel, and the disk drive will run itself. It essentially is another computer, a 6502 based one, to be exact, that only accepts commands from a serial line and performs all of the disk commands. Imagine.. a computer so STUPID that you need TWO computers to run any disk software...and you would be paying for TWO computers also. Commodore doesn't tell the average users that they are essentially using TWO computers instead of one. Apple computers are also based on the 6502 series of microprocessors. Apple used an old method of running it's computers...just write a DOS and put it on disk, and when the computer is powered up, the DOS is loaded. Funny thing, though. Although Apple boasts of 64k of RAM, much of that is used to hold the resident DOS, and BASIC. If you want to load a program which needs the space allocated by DOS, you are out of luck, since your program might make DOS calls to perform disk functions. And if DOS were overwritten, then when your program is finished, it must go back and load it all over again. And 6502 is not exactly the processor I would waste terribly expensive memory on, since it has a very limited instruction set, and things I take for granted now, like memory-to-memory word moves, multiplication, division, and subroutine branching would be terrible to implement on an apple of commodore 64. I just don't know how they have survived this long...

Our little TI, on the other hand, has a wonderful method for handling new devices. The GROM header, present on all ROM in the expansion box, and all command modules, is the link between the unknown and the known. It allows us to plug in new devices at any time in the future, and the operating system will immediately recognize the device, as if it were there from the beginning. This is what will keep our TI computers alive. The method of access is very similar to the IBM pc method. Each peripheral card has a certain address in the serial addressing fields. The operating system can turn on a card singly, look at what occupies a pre-defined memory area (>4000 to >5FFF for us) and can determine if the device exists. With the IBM, certain logical names are assigned to a physical device address, such as COM1:, TTY:, A:, LPT1:, and so on, and can be changed according to the user's wishes. This requires a small modification to DOS to accommodate the new device, and from then on, a new sub-version to dos is created. If the device is removed, an error will be issued since DOS can no longer locate the installed device.

The GROM header in the TI provides a standard table for finding a device quickly and efficiently. All of the devices use a pre-decoded 8k block of memory, and 8k is plenty for most devices. Since we are not limited to

TACOMA 99ERS
USERS GROUP
P.O. BOX 42383
Tacoma, WA 98442

DISCLAIMER

MEETINGS ~ 1st and 3rd THURS. ~ 8 p. m.

South End Pool Building - 402 E. 56th Street - Tacoma, WA

For More Information Call 474-7310