22 JUN 87

Topics

LA 99ers COMPUTER GROUP Newsletter

VOL 6 NO. 6 JUN. 1987

TnT

THE FEST THAT WAS

Caring and sharing were the most obvious by-products of Fest-West. We very much follow the Avis slogan "WE are number 2, so we try harder." It is not a contest of the most attendees, or the most vendors. It is just being there. The success is the effort of many. None of us can be all things unto all people. Setting aside ego and being open to anything does away with tunnel vision. We strive to improve by admitting imperfection. B.J. Mathis of Tucson found a need and not only willingly stepped in to handle it, but already has expressed her availability to S.N.U.G. (Las Vegas) for Fest-West 88 registration. Our personal thanks B.J. for bailing us out.

There is a song around now "I've got a new attitude." That is something we all can consider. The months preceding the Fest, Gail Fair (mostly) and Terrie visited other User Groups encouraging participation. There is a vast difference in attitudes and emphasis among these groups. We can all learn from both the negative and positive aspects observed. Case in point, the groups that have more hands on participation by their members during meetings are the ones with the least difficulty in getting group cooperation in events such as the Fest. We observed this in Boston with the excellent group participation of the Nutmeg group, small but very active. Here we saw similar enthusiasm and cooperation among the Brea group. Gail has been very impressed with the quality of their meetings and we will take their lead. Demo only meetings can indeed lead to inertia. We in the last year or so have attempted to invigorate our meetings, and have clearly done so but see yet more room for improvement. The fruits of Brea's labor were in the number of new members signed up at the Fest. Their booth was fun and an asset to the Fest. Thanks.

The announcement that 99'FEST-WEST'87 would be held in Las Vegas has certainly brought about some real positive communications. Cheryl (Regena) Whitelaw is just bubbling over with enthusiasm and willingness to participate in a hands on manner. Great news for SNUG. I just called John Martin and passed along this happy information. Thanks Cheryl. The date for Las Vegas is still up in the air, they are attempting to get the schedules of other fests to avoid a conflict.

Over the next few months we will be printing ads from the various participants in the Fest. Their products are all worthwhile and bear your consideration. Many will also be available through our MARKETPLACE at discounted prices.

Just a little dry information on the Fest nitty-gritty, only because mis-infomation is circulating. Other area Fest holders can attest to hidden costs, rental, insurance, electricity, decorating (booths, tables, chairs). It all adds up. We sub-let the Shrine Mezzanine in conjunction with an ongoing main level computer event. We are not bothered with the detail work, only meeting the cost of the sub-lease. The cost was a 50% increase over last year and was very obvious we could not carry it alone. Tom Irwin a former LA 99er and now President of the SCAN Amiga group was approached by me to split the mezzanine and the lease cost. This was successfully accomplished. It was necessary to increase the booth rental over last year to avoid going too far out of our treasury. $100 for two days is commensurate with $50 for one day charged by other areas. We do not get the gate cost, the promoter gets that, deservedly. There is a definite advantage to the generic computer related values on the main floor.

A couple of User Groups sent a donation to fund the Fest, the balance was from the LA 99er treasury. The telephone line installed for the GEnie international conference, the hospitality room, beverages and snacks, the Fairware booth and mailings, the 1500 Postcard notification are some big ticket items we sponsored. Early incomplete estimates show a potential ncn-recompensated cost to the club of $500.00. We feel it was all worthwhile.

BUYER BEWARE ALERT

We have recently had two communications from international members about their experience with Pilgrim's Pride. Apparently they sent funds for merchandise and have still not received anything. Furthermore, their letters to the company have gone unanswered, although their checks were cashed. Until this matter is cleared up, we would suggest not dealing with this company.

June 24th meeting, we will have a great double bill, our own Doug Moore on FRACTALS. (if you don't know what they are be ready for a big surprise). BAT our new correspondent, Barry Traver will show us what we missed in LA by his being at the excellent Ottawa Fest.

BE THERE!!!

THE BEST THAT WAS

Our User Group exchange list is close to 140, in and out. We read every one of them. There are some gems there, and then there is a disturbing trend also. We have always been candid, by volume of comments more often appreciated than not. To the man we react not act. There has usually been provocation, scams and unproductive action. We receive as well as give, usually tell it as we see it and Caveat Emptor. Recently there has been an outpouring of unfounded venom and judgmental criticism unfairly spewed. I (Terrie) do the best I can do at any given moment. Those that know me well know it is within the framework of a very severe non-functional depression. I apologized some time ago by mail to an individual for my "procrastination", the reply was "I had no right to procrastinate"! Well with that as an example of standards set up for me by another, I will continue this discussion of "best".

The best that Richard Mitchell can do with his fine Smart Programmer is not good enough for those who demand it be exactly on time, and accuse him of cheating them ala HCM. Richard is a full time employee with an excessive amount of expected overtime, he is a husband and parent with at least the same amount of domestic stress lots of us have to deal with. All this in addition to singlehandedly compiling a fine publication. Why is this not recognized as the best he can do at this time.

Barry Traver, his sharing of himself on both Compuserve and GEnie give him an extra dose to be aware of his best not being enough. It indeed does pain him that the remarkable Genial Traveler is not as timely as he had hoped. Just in the last month he felt compelled to spend an incredible amount of postage money to let us know an issue was imminent, just to follow it up with an errata notice. Anyone with paper and pencil can calculate the cost of 9 disks, 6 mailers, two postcards per subscriber and come in way over $30.00 subscription fee. In addition to the two data networks, Genial Traveler, Genial Computerware, Barry home-schools John Calvin and Preaches. Where is his best not enough?

Craig Miller, his best was a red flag challenge to a certain mentality to break and circulate. These then have their defenders ad nauseum, those who publically stand up for Craig are called irrational Miller worshippers. Really! Now a certain few are using words like "cheat" "steal" "deserter" "traitor". Based on what? Not facts that is for sure. Turbo-XT was Tritons idea, not vice versa. It was geared to the console and tv set, (by far the majority). Deserter? just who do you think designed the new fantastic Super Extended Basic module.

Step down off the vituperative soap box and recognize the destructiveness of this parochial pontificating. Recognize the unsung productive persons within your group, be they young or old, sophisticated or down home, encourage and expose them. Is your glass house without streaks? We can all learn from one another. I can't (or haven't) program, but I sure can appreciate Tom, George, Craig, Doug, Mike, Peter, Barry, Richard, Chris, etc. I am enriched by them, they produce action not drivel.

No one above has asked me to publicize their linen, me neither. It is just the other side to the rank criticism rampant now. Blow the whistle on deservers, but for goodness sake get your facts straight and be honest, stash the ego.

Additional note from Tom: I am in completel aggreement with Terrie's sentiments expressed above. I just would like to add, with regard to my "open letter" a couple of months back, that I continue to maintain the policy of not criticizing anyone who has not gratuitously "bashed" someone else first. I defend my friends and those I respect, not because of "hero-worship," of which I have been accused, but rather because I do not choose to associate with those whose own ego gratification seems to depend on nastiness towards others. Enough said - I tend to get rather emotional about the subject, as you all know!

WANTED for Schoolteacher....
Word-Processor not requiring disk system.
EX Okay. Sol Shulman 213-375-9840

WANTED - Manuals for following modules:
PYRAMID PUZZLER, STARMAZE, FRACTIONS2,
and READING RAINBOWS
Joe Fierstein 213-377-9834

BAT'S "IN THE BELFRY"

"Twinkle, twinkle, little bat!
How I wonder what you're at!
Up above the world you fly,
Like a tea-tray in the sky."

Perhaps you recognize those words by the Rev. Charles Lutwidge Dodgson, mathematics don, clergyman, and literarian. Since I share those (a)vocations, perhaps those words may also serve as an introduction to this column, "In the Belfry," written by Barry A. Traver a.k.a. "BAT" (watch that "tea - tray-ver is the correct pronunciation, long ay, no ess).

This column will have no consistent theme from issue to issue, so you will indeed be able to wonder what I'm at! (In fact, we may make this place a real wonderland, which is a world within itself, for all is in wonderland, so to speak.) But enough of introductions. In this issue I'd like to mention (promote?) two items that should be of interest to many: (1) a "sort experiment" by J. Peter Hoddie and (2) a source/code tutorial by Wayne Stith. The first is fairware, the latter is public domain, but we are I believe much in debt to both authors.

(1) J. Peter Hoddie's SORT/EXP is an assembly language sort program that will in essence sort any type of file of any record size (DV8Ø, IF2Ø, or what have you) on up to 8 different user defined fields. It can handle either 1ØØØ records or 24K of data (whichever comes first) and sort in either ascending or descending order, using either a shell sort or shuttle sort (your choice). It is very fast, perhaps twice as fast as the famed Clulow/Romer sort (which, however, was limited to DV8Ø files).

The program is very easy to run. You supply the name of the disk file you want to sort, and then indicate the starting character position and length of field of each field you want sorted. (Most people will not need to sort on eight fields, so you can just tap <enter> when you've provided information for the fields you do want to use.) After you tell the program whether you want ascending or descending order and which type of sort to use, the only thing remaining is to supply the name of the disk file to which you want the sorted data to be written.

There's one bit of information not included in the present documentation which you should know: If you make a mistake in typing a file name, you can correct your entry by using CTRL-H (FCTN-S will not work). The program was originally written on a Myarc 964Ø (the "Geneve"), and CTRL-H and backspace are equivalent on the Geneve keyboard (and on most computers, for that matter).

The first release of this program - sent out on the first volume of the John Calvin fairware/public domain project - contained a minor error in the source code. (Unless you entered other than a single-digit number for starting character, you would never notice anything wrong.) Here's what to check in the source code for the file SORT/EXP/S. Lines 72Ø-722 should read as follows:

```
Ø72Ø    SRL  R3,8         make it a word
Ø721    MOV  R2,R1        Thank you, Tom Freeman!!!
Ø722    MPY  @D1Ø,R1      multiply current value times 1Ø
```

If you have a copy of SORT/EXP/S that does not read like this, just insert the MOV R2,R1 before the MPY command, and you'll be all set.

Here's how to make the program image SORT/EXP file, if it hasn't already been done for you. Put the source code on a disk in drive one. (Yes, Virginia, Peter does give you all the source code!) Assemble SORT/EXP/S to create a file that you can call SORT/O. Then assemble SAVE/S (also supplied by Peter) to to create a file that you can call SAVE/O. Then - using option 3 of the Editor/Assembler - load in DSK1.SORT/O, load in DSK1.SAVE/O, and link to program name SAVE. You have just created DSK1.SORT/EXP, a program image (Editor/Assembler option 5) version of the sort utility!

How do you obtain a copy of this exceptionally useful program? Individuals can freely copy it and pass it around. User groups or commercial organizations must first obtain written permission from the author for permission to distribute, but you can download it from CompuServe or GEnie (if you have a modem) or order it for $3.ØØ from John Calvin Traver, 835 Green Valley Drive, Philadelphia, PA 19128 (cost includes disk, copying, mailer, and first class postage). If you like the program, you can send $1Ø.ØØ to J. Peter Hoddie, 12 Paul Revere Road, Lexington, MA Ø2173.

(2) Wayne Stith's public domain program KWIKFONT appeared in the Genial TRAVelER #6. The program is a character pattern editor, by which one can alter character patterns and store them on disk for later access and use by a BASIC program. The program is self-contained, requiring no external documentation, and - as you might expect, since it's all in assembly - is very fast. (Genial TRAVelER also contained a utility to create a CHARA1 file from a character set created from KWIKFONT.) What many people don't know is that (a) the full source code can be ordered directly from Wayne at a nominal price ($6.ØØ) and (b) the source code is actually - in my opinion - one of the best tutorials currently available for learning assembly language.

The tutorial takes up both sides of a flippy, and prints out to about seventy pages or so. You'll find things clearly explained here that you won't see explained clearly (or often at all) in Lottrup, McComic, Molesworth, or Morley. If you're a novice in assembly language, you'll find a lot of help in this material. (I have to admit that Wayne helped me get a handle on a number of things I wasn't sure about myself!) If you want a copy, just send your $6.ØØ to Wayne Stith, 715 Timken Drive, Richmond, VA 23229. I don't think you'll regret it. (Wayne calls it "an idiot's guide to assembly language," but that's exactly what some of us have been looking for!)

Well, that's enough for this month's column. And - if you guessed that the Reverend C. L. Dodgson used Lewis Carroll as a pen-name and is the author of Alice in Wonderland, from which the opening quote came - you are to be congratulated. ("I wonder what he's going to come up with for the next issue....")

FROM THE DISK OF....

==== === ==== ======

by Mike Dodd

Yes, I did write a column last month. What, you didn't see it?? Gee. Now that I think of it, neither did I! We have the wonderful (?) US Postal Service to thank for that. I mailed it out May 8 "Next Day Express Mail," and Terrie got it the 11th. Very impressive. Oh well. This column, therefore, is last month's column with some of this month's thrown in places. Terrie has threatened me with a horrible punishment if I don't write something, so you can send hate mail to her for making you read this.

The Dots Perfect, which I received as a gift from Terrie, Tom, and George, is a wonderful device. I've got one for my yuccky old TI MX8Ø printer, and the print quality has improved dramatically. It gives my printer, in addition to what it already had, optional slashed Ø's, italics, super/sub script, and NLQ (Near Letter Quality) mode. PLUS, it lets me pick the font settings I want from the printer, instead of having to send a control code to it. For NLQ, I hit FF with the printer on-line. For draft, it uses LF. To activate the menu, I press the On Line and FF keys together. Now I go through the menu by pressing the FF key, and toggle the choice on or off with the LF key. When I'm done, I press On Line, and the printer is ready to print. The menu choices are: Condensed, Enlarged, Emphasized, Double Strike, Perferation Skip, 1/2" left margin, Italics, Underline, Fine Print, 8 Lines Per Inch, and Slash Ø. Fine print is a combination of Condensed, Superscript, and 6/72" line spacing (12 LPI). That is TINY print, but VERY legible.

All in all, I'm very happy with Dots Perfect. If you have an older model printer, this is definitely for you. Note that it comes in models for many different types of printers. I have only one complaint with Dots Perfect: it doesn't include Elite mode, which I had on the TI printer with the Epson EPROMs installed (no, it wasn't documented, but it was there). I'm surprised that Elite isn't included in this. But still, it's worth it, and don't let the loss of Elite stop you (I'm not sure if all the models of Dots Perfect lack Elite - the versions for other printers may have it).

Here is a fix to one of TI-Writer's "features" (aka "bug"):

Using a sector editor program (i.e. Advanced Diagnostics), edit sector 3Ø (31 if counting from one) of the FORMA1 file (or the FORMAT file with My-Word). At byte 58 (hex >3A), change the >AØ67 to >AØØ2. This will stop the formatter from automatically putting two spaces after a period.

There are other changes I have made to the Formatter as well. I have changed the @ and & (bold and underline) to ` and \ (FCTN C and FCTN Z), and I changed the "*" (for mail merge) to a "|" (FCTN A). The reason I changed the last is that, in a document, if you have an asterisk followed by two or more numbers, such as *11, or *256, the formatter will drop the * and the two digits immediately after it. That's what happened a few places in Tom Freeman's catalog program. Anyway, the changes are all in the first sector of the FORMA1 file. For the @ and & fix, change the @ and & at byte 115 (hex >73) to ` (FCTN C) and \ (FCTN Z). or whatever you want them to be. To change the "*" to "|", change the * at byte 112 (hex >7Ø) to | (FCTN A), or anything else. Rick Cosmano of the SCCG is to be credited for the @ and & fix, the other two are mine.

By the way, the change for @ & * is totally useless on the Geneve's MY-Word processor, since the formatter for it includes commands to change them.

Terrie Masters' big problem is that she is too modest to acknowledge the fact that several people think she is an incredible person. Since we obviously can't let that happen, I'm taking it upon myself to remedy the situation. After the Fayuh in Boston (April 4), a lot of us went to a ballroom at a nearby hotel to talk. There, the Ottawa Users' Group, which had 16 members there, presented an award to Terrie for her outstanding support of the TI community. Congratulations, Terrie! You deserve it. (The award was what Terrie was referring to in her column when she said "Thank you Ottawa, what a surprise.")

By the way, at the May 16 Ottawa Fest, Barry Traver was given a similar award for his support of the TI community. Congratulations to both of you!

A few people have asked how to detect the differences between different versions of XB and different peripheral cards when writing a program. The method I used in XBasher to detect MYARC XB II or TI XB (or one of its offshoots, i.e. Mechatronic, GK XB, etc) was:

```
    1Ø CALL VERSION(X)
    2Ø IF X=12Ø THEN RUN "Triton Super Extended BASIC
version"
    3Ø IF X>=3ØØ THEN RUN "Geneve 964Ø Advanced BASIC
version"
    4Ø IF X>=2ØØ THEN RUN "MYARC Extended BASIC II
version"
    5Ø RUN "TI / GK Utility I / Mechatronics Extended
BASIC version"
```

To detect between a CorComp and TI disk controller card, the method I used in my modification of REDISKIT was:

```
    1Ø ON ERROR 4Ø
    2Ø DELETE "LD-CMDS"
    3Ø RUN "CorComp version" 4Ø RUN "TI version"
```

I don't know an easy method (or even a hard method) to detect the MYARC controller. Does anyone out there have any bright ideas?

Last month, George Steffen wrote an article on eliminating memory waste from assembly programs. Over the years, I have come across a few tricks for XB, as well.

One programming method I frequently see is:

```
    1Ø PRINT "HELLO" :: PRINT "HOW ARE YOU" :: PRINT "I
AM FINE"
```

That can be drastically shortened to:

```
    1Ø PRINT "HELLO":"HOW ARE YOU":"I AM FINE"
```

That does the same thing, only quicker and uses fewer bytes.

Another programming pitfall is the line:

```
    1Ø .....
```

```
20 IF X=A THEN 30 ELSE 10
30 .....
```

Line 20 can be redone to:

```
20 IF X<>A THEN 10
```

Again, it runs quicker and takes less memory. Here are the opposites of several relational operators:

= is the opposite of <>
< is the opposite of >=
> is the opposite of <=

So, if the relational clause in line 20 was <=, then you could change it to > and eliminate the ELSE.

Here's one example of where XB can save many bytes. Take the following:

```
10 A=2 :: B=2 :: C=2 :: D=2
```

This can be changed to:

```
10 A,B,C,D=2
```

The same goes for string variables. A$="" :: B$="" can be changed to A$,B$="".

Another thing I've noticed (and one that I've been guilty of, too) is the practice of putting, at the start of a program, a command to set numeric variables to 0 and string variables to "". When XB starts to RUN a program, it automatically sets ALL numeric variables to 0 and string variables to "" (null string), so there's no need for you to do that. The only exception to that rule is if you use the RUN "filename",CONTINUE command of MYARC XBII or Advanced BASIC, in which case the variables are not changed.

Finally, here's a very small way to save bytes (or byte, in this case). Take the following:

```
10 CALL KEY(0,K,S)
20 IF S<>1 THEN 10
```

As you'll recall, the Status from a CALL KEY will be set to either -1, 0, or 1. So, if S is not set to 1, S will be less than 1. Therefore, line 20 could be changed to:

```
20 IF S<1 THEN 10
```

There are more ways than what I've put here to save bytes in XB, these are just the ones I can think of off the top of my head.

For the "Mike Dodd almost-useless program of the month", I present to you an assembly program designed to act as a stationary TRACE command. It displays the line number at the bottom of the screen, without scrolling. I used this when writing XBasher to detect which lines were taking the longest to execute. Credit must be given to J. Peter Hoddie for giving me the idea for this, and for providing assistance with one of the bugs. Thanks Peter!

After I wrote my TRACE routine, I decided, in a fit of madness, to fully explain and comment the program, so that others might learn something from it. I hope that my comments actually help you, rather than confuse you.

To use TRACE, type it in with the Editor/Assembler. Assemble it with the object file named TRACE/O and with the R option.

To use the program, go into Extended Basic and type CALL INIT :: CALL LOAD("DSKn.TRACE/O") <ENTER>. Now load your XB program. To turn on the trace, type CALL LINK("TON"). To turn the trace off, type CALL LINK("TOFF"). You may also use these commands from within your XB program. If you have the trace turned on when you type RUN, you will see the line numbers in the program quickly scan by on the screen before the program actually starts to do anything. These are the line numbers that XB is pre-scanning.

The comments in the source code give directions for changing the row and column for the position of the trace.

If you have any questions, write or call me (my address is in the source code).

```
* TRACE ROUTINE
*  Copyright (C) 1987 by Mike Dodd
*                        116 Richards Drive
*                        Oliver Springs, TN 37840
*                        615/435-1667
* Finished on April 28, 1987

* Stationary trace. Prints line number to screen.
*
* Set Row and Column here
* WARNING: DO NOT set these outside of their limits
* (1-24 for row, and 1-32 for column). As the
* program does no error checking, it could cause
* unpredictable results if you set them out of
* their range.
ROW    EQU  23          Row 23
COLUMN EQU  11          Column 11

* Use CALL LINK("TON") and
*     CALL LINK("TOFF")
*  to turn trace on and off.

       IDT  'MIKEDODD'

       DEF  TON
       DEF  TOFF
TOFF   CLR  @>83C4      wipe out ISR
       RT               return
TON    LI   R0,TRACE    set ISR
       MOV  R0,@>83C4
       RT               return

* buffers
SUBWS1 BSS  >20         Subroutine WS
MYWS   BSS  >20         Main WS
SBUF   BSS  6           ASCII for line number
* LINE stores the last line number the program
* printed. The reason for saving this is that if
* the line on the screen is the same as what it is
* now ready to print, there is no reason to print
* it again, which saves time. If the program
* printed the line number every interrupt (every
* 1/60 second), XB would be greatly slowed down by
* this program. As it is, it is barely noticeable.
LINE   DATA >FFFF       Stores last line # printed

* Main routine
TRACE  LWPI MYWS        Load my WS
* In XB, >8344 stores a one byte flag. If this
* flag is equal to >00, that indicates that we
* are in command mode, so we don't want to check
* the line number. If the flag is equal to >FF,
* then a program IS running, and we do want to
* check the line number. Note that this flag will
* say >FF on Pre-Scan time as well, so this routine
* will print the line numbers it is scanning, as
* well.
       MOVB @>8344,R0   Are we running a program?
       JEQ  RETURN      No - go back
* In XB, >832E points to the second word in the
* line number table. The line number table is
* arranged with the line number first, then the
* start address for the line. Since >832E points
* to the start address, we need the word 2 bytes
* before it. Also, since the line number might be
* stored on an odd byte, we can't just use the MOV
* command, since it only works on even bytes. So,
* first we will get the value at >832E and store it
* in Register 1. Then we'll DECT it, which
* subtracts two from it (R1=R1-2). Then we'll move
* the first byte to the Most Significant Byte (first
* byte, or MSBy) of R0. Then we'll swap R0 so that
* the first byte is moved to the second, and the
* second to the first. Then we'll get the second
* byte of the line number. Now, R0 has the line
* number, but it's backwards - second byte first,
* first byte second. So, we'll swap it back with
* the SWPB command, and it will be fixed.
       MOV  @>832E,R1   Get pointer to current line
       DECT R1          -2: Back up to line number
       MOVB *R1+,R0     Get MSBy
       SWPB R0          Swap bytes
       MOVB *R1,R0      Get LSBy
       SWPB R0          Swap back
* See if this is the same line as the one we
* printed last time.
       C    R0,@LINE     Same as last time?
* If it is equal, the EQual bit in the status byte
* will be set. The JEQ instruction means "Jump
* if EQual bit is set." So, if they are equal, then
* it will jump to RETURN.
       JEQ  RETURN       Yes - return
* We want to be sure to save the line number, so we
* can compare it next time.
       MOV  R0,@LINE     Store
* Make R1 point to the start of the buffer for the
* line number.
       LI   R1,SBUF     Point to buffer
* Now R0 is the number to convert to ASCII, and R1
* points to the buffer. These are the requirements
* of the BLWP @CNS subroutine
       BLWP @CNS        Convert to ASCII
       LI   R0,ROW*32+COLUMN-33  Screen location
       LI   R1,SBUF     Buffer
       LI   R2,5        Five digits
       BLWP @PBASIC     Print with BASIC offset
* Now it's time to return to Basic. All we have to
* do is load the GPLWS, and do a ReTurn, since R11
* wasn't destroyed by us.
RETURN LWPI >83E0       Load GPLWS
       RT               Return

* Print with BASIC offset
*  In: R0=VDP address to print text
*      R1=CPU address of text without BASIC offset
*      R2=length of text to print
PBASIC DATA SUBWS1,PBAS1 WS to use & start address
PBAS1  MOVB *R13,R0     Get MSBy of VDP address
       MOVB @1(R13),@>8C02   Move LSBy to VDPWA
       ORI  R0,>4000    Set for write to VDP
       MOVB R0,@>8C02   Move to VDPWA
       MOV  @2(R13),R0  Get R1 (CPU address)
       MOV  @4(R13),R1  Get R2 (length)
PBAS2  MOVB *R0+,R2     Get byte
       AI   R2,>6000    Add BASIC offset
       MOVB R2,@>8C00   Write to VDP
       DEC  R1          Are we through yet?
       JNE  PBAS2       No, keep going
       RTWP             We're done - return

* Convert number to decimal string
* IN: R0=Number to convert (from 0-99999)
*     R1=CPU location to put five digit string
CNUM   DATA 10000,1000,100,10,1
CNS    DATA SUBWS1,CNS1  WS to use & start address
CNS1   CLR  R2          Pointer to divisor
* If R3=>0000, then all we've gotten is 0's.
* If R3=>FFFF, then we've gotten to a non-zero.
* The reason for storing this information is so that
* the routine doesn't print leading 0's. If R3=0 and
* the byte it's ready to print is 0, then it will
* print a space. If the byte is 0 and R3=>FFFF, then
* we're in to the numbers and we do need to print
* it. If we didn't do this, the number 805 would
* come out as 00805. If we just told it not to print
* any 0's at all, that number would come out as
* "8 5", which doesn't really signify 805.
       CLR  R3          Haven't reached a non-zero
       MOV  *R13,R5     Get the actual number
       MOV  @2(R13),R1  Get addr to put string
* The DIV X,RY instruction performs the following:
* RY = RY & RY+1 / X
* RY+1 = remainder
* In other words, if R0 was 3, R1 was >0001, and
> R2 was >86A0, the DIV R0,R1 instruction would
* divide 000186A0 (decimal 100,000) by 3, setting
* R1 to >8235 (decimal 33,333) and R2 (the
* remainder) to >0001.
CNS2   CLR  R4          The first two bytes are 0
       DIV  @CNUM(R2),R4 Divide by 10000,1000,100,10
*                                 or 1, depending on
*                                 what R2 points to.
       MOV  R3,R3       Have we gotten to non-0's?
       JNE  CNS4        Yes - print the character.
       MOV  R4,R4       Is this a 0?
       JEQ  CNS3        Yes - print a space.
       SETO R3          No, so set R3 appropriately
       JMP  CNS4        And print the character
CNS3   LI   R4,>20      character for space
       JMP  CNS5
CNS4   AI   R4,>30      Add ASCII offset to it
CNS5   SWPB R4          Put in MSBy
       MOVB R4,*R1+     Move to buffer
* Note that after the division, R5 will have the
* remainder, which is all we want now anyway.
* For instance: suppose we want to convert the
* number 25000. First we divide by 10000 - we
* get 2 for the quotient, and 5000 for the
* remainder. Next we divide by 1000 and get 5 for
* the quotient, and 0 for the remainder. If we had
* used the whole number both times, we would have
* gotten 2 for the first digit, and 25 for the next,
* which wouldn't work, since we can only use single
* digit numbers.
       INCT R2          Point to next
       CI   R2,10       Through yet?
       JNE  CNS2        No
       RTWP             Return

       END

```

XBASHER, A Review

by Scott Darling, (C)opyright 1987

Written by Mike Dodd, Distributed by Genial Computerware, Box 183, Grafton, Ma. Ø1519, $1Ø.ØØ, All A+'s.

This program is needed by anyone and everyone!! No clarification you say?? EVERYONE has an Extended Basic program! AT LEAST one!! This program will make that one program run faster and reduce its size. GUARANTEED!! Most of us who have been around the Ti World for awhile remember what SMASH is. The BAD part about SMASH is you had to start it at night and HOPE it was done by morning!! You won't have to worry about XBASHER! Xbasher runs out of the Extended Basic environment. There are two versions available. One for TI XB and one for Myarc XB II. No mention was made of the 964Ø compatibility. Probably because the 964Ø will be so much faster. You can even run XBASHER on combined XB and A/L programs. Complete instructions are given on how to do this!

To run XBASHER requires that you save your Program in Merge format using the following: "OLD DSKn.filename" then "SAVE DSKn.mergename,MERGE". Then insert the XBASHER disk in drive and select XB. The disk files will determine which XB you are using and load the correct version of XBASHER. After the program has loaded, you are presented with a title screen. Next is the option screen. Which is: Shorten Variables, Crunch Lines, Remove REMS and !'s, Remove Let's, Change CALL CLEAR to DISPLAY ERASE ALL (this one alone saves you 5 Bytes!), Don't Change CALL SUB routine Digits, and Change Constants. Some of these are obvious as to what is going on.

Shorten Variables will take all your String and Nonstring Variables and shorten them to one then two character variables. There is an immense saving in memory by doing this. Tho, most people like to have a 'name' for variables. If the variable name is less than 3 characters it is no saving in memory. Its when you go over this limit that memory is being eaten away. There is also an option to print the Variable list to an output device.

Next is crunch lines. This was VERY impressive. XBASHER will crunch or combine lines together. So what about the lines that are GOTO'ed you ask?? (Well somebody will ask!!) The A/L in XBASHER keeps track of the logic flow of the program! THIS part makes the program FAR superior to SMASH!! The only bad thing about this function is that the line length of a line number is so long you may not be able to edit the new line!! Considering this is the only drawback, it is a worthwhile option! I have been able to get 8 lines of code to a line number.....so did XBASHER. Next is REMOVE REM's and !'s. Remarks are good for developing a program but are a hindrance when actually running the program. This option will delete them and restructure the resulting deletion of them. Remove LET's. PLEASE I hope everyone by now realizes the LET statement is inconsequential to programming!

Change CALL CLEAR to DISPLAY ERASE ALL. Nothing irks me more in XB programs than to see a "345 CALL CLEAR" then "35Ø DISPLAY AT(12,1):...". If you use "35Ø DISPLAY AT(12,1)ERASE ALL:" it does the same thing as CALL CLEAR and saves memory!!

Next is Don't change Sub Digits. What this option does is change the numeric constants to the characters @,\,[,],and_. This saves 2 bytes per each occurance of the variable. But, because of the nature of CALL SUB routines this may cost you MORE memeory than any savings. Also, note. CALL SUB routines are like a separate XB program within a program. Consequently you can use identical variable names in CALL SUB's as in the program without any type of error received by the Basic Interpetor. Also, CALL SUBS are slower processing than GOSUB's. The only advantage is to CALL SUB's is variable variable passing!! (Are we confused yet??) Lastly is the Change Constants option. Basicaly what was said in the previuos paragraph applys to this option. EXCEPT in this environment, this option will save you memory. Don't ask me why there is a difference. Just believe me!! So much for the option list. Each option has a letter reference. By pressing that Letter toggles each option on and off. Hitting X says you like what you see on the Screen.

Next screen asks for the input file name. The one you saved in MERGE format and checks to see if you remembered the filename correctly. Then asks for an output name. And even provides a suggested name. Next is an output device and name for the variable listing if you selected that option. FINALLY the computer starts doing the work!! The screen will show you the status of the program. A line count, the last line number referenced by a goto, gosub staement will be shown on the screen. Xbasher makes two passes thru a program. First to make lists of variables, line numbers and other info. The second pass will write the new program to disk. How long will it take?? The size of the program involved is the ONLY factor. I ran an 11 sector file thru XBasher and it took 5 minutes to do the job. The savings were 5ØØ bytes. Next I ran the ultimate EGO test on XBASHER. I wrote a BBS program that is 9Ø plus sectors long. Almost 23K in bytes. So, I ran XBASHER against it. I felt I was a decent XB programmer and there was no way XBASHER was going to save any bytes in MY program!!

Well after about 3Ø minutes and my selecting ALL the options. The darn program found 2ØØ bytes somewhere!! I'm still trying to see where it found them!!

To sum it up, Xbasher is the perfect compliment to any XB program. You only need to run it once, and save the resulting code. XBASHER will show you what XB programming is all about! There is a lot of power in that cartridge!!

XB:BUG, A Review

======= = ======

by Scott Darling, (C)opyright 1987

XB:BUG is written by J. Peter Hoddie and Distributed by Genial Computerware XB:BUG is an unusual program. It is like DEBUG for the Editor Assembler. It can be resident in memory and called upon at any time. It allows you to follow a program as it progresses through what YOU programmed. As an experienced XB programmer, I can't tell you how many Hours I spent MANUALLY tracing, deciphering, and endless mapping of a program to see where I went wrong!! We know have such a program to take all the FUN(?) out of the old methods. XB:BUG requires XB version 11Ø and above, Disk, and 32K. Printer is optional tho almost essential. I also ran XB:BUG on the new Triton Superxb with no problems. XB:BUG will NOT work with Myarc XB II. as the memory locations are totally different.

The only limitation to XB:BUG is not the program but the memory limitations of XB. The program is 5K long. If you attempt to use XB:BUG on hybrid XB/AL files. You will have to remember that XB:BUG loads in Low Memory. There is a >AØØØ version on the distribution disk. You will be limited to and 18K program in the >AØØØ space. Like I mentioned, this is not a limitation of XB:BUG, but the 4A's.

To load XB:BUG, use the standard "RUN DSK1.LOAD" format or auto boot from power up. Normally you would load XB:BUG first, If, for some reason, you want to load XB:BUG after your code, there is a version on the distribution disk. After XB:BUG is loaded the READY prompt will return on the screen. Now you can load your XB program in the normal method. To activate XB:BUG you press the Control and Shift keys simultaneously, or you can do a CALL LINK("GOBUG"). Then you will be presented with the main debugger prompt. The following commands are available. Array: This allows you to inspect the contents of an string or non string array.

Breakpoints: Setting breakpoints allows you to stop the execution of the program at various points to check for the other functions of XB:BUG. It is NOT necessary to do this in all cases. But, sometimes the program may execute too fast for XB:BUG to literally catch what you want to examine.

Change: This allows you to change the value of any numeric variable. You first have to invoke a V or A commands.

Data: This gives the line number from which the next READ will get its DATA and also shows the next actual DATA item that will be read.

Files: Lists the unit number and device name associatted with each open file. The "mode" of the file was opened in is also given. Input,update,append, or output. Any data in the I/O buffer will be displayed.

Graphics: This item gives you information on 3 items. 1) Character definitions. 2) Color Definitions, and 3) Sprite status. You can manipulate all 3 items.

Kill Sound: This turns off the sound chip. You will like this after going back and forth from XB:BUG and XB.

List: Will list the program you are working on. You can set the line numbers you want to list.

Other Variable Space: This item is a beauty. It allows you to inspect variables in the main program AND also in Subprograms. This one is complicated to explain so read the manual!

Program: This supplies information about your program. Line number executing, ON ERROR line number, and OPTION BASE. Also, On BREAK, and TRACE if they are active.

Quit: Quits XB:BUG.

Subprograms: Lists all defined subprograms.

Trace: This will trace back all pending GOSUB and SUBPROGRAM returns.

Variables: List variables and functions with their current values. If there is an array, it will list the DIM. This also works on Subprograms. ?: Will list a line oS valid keystroke commands for XB:BUG.

Math functions: Allows you to perform simple calculations.

Match Function: Several of the commands in XB:BUG will prompt for a MATCH string. The one thing I will explain about this concerns the manual. It says you can use a wildcard character. Well, the printer made the Asterisk so SMALL you may miss this in the manual! I DID!! The quotation marks and the asterisk combined to make a nice inl blob to my old eyes!!

This is all of the commands. The manual documents each command far more than I have here.

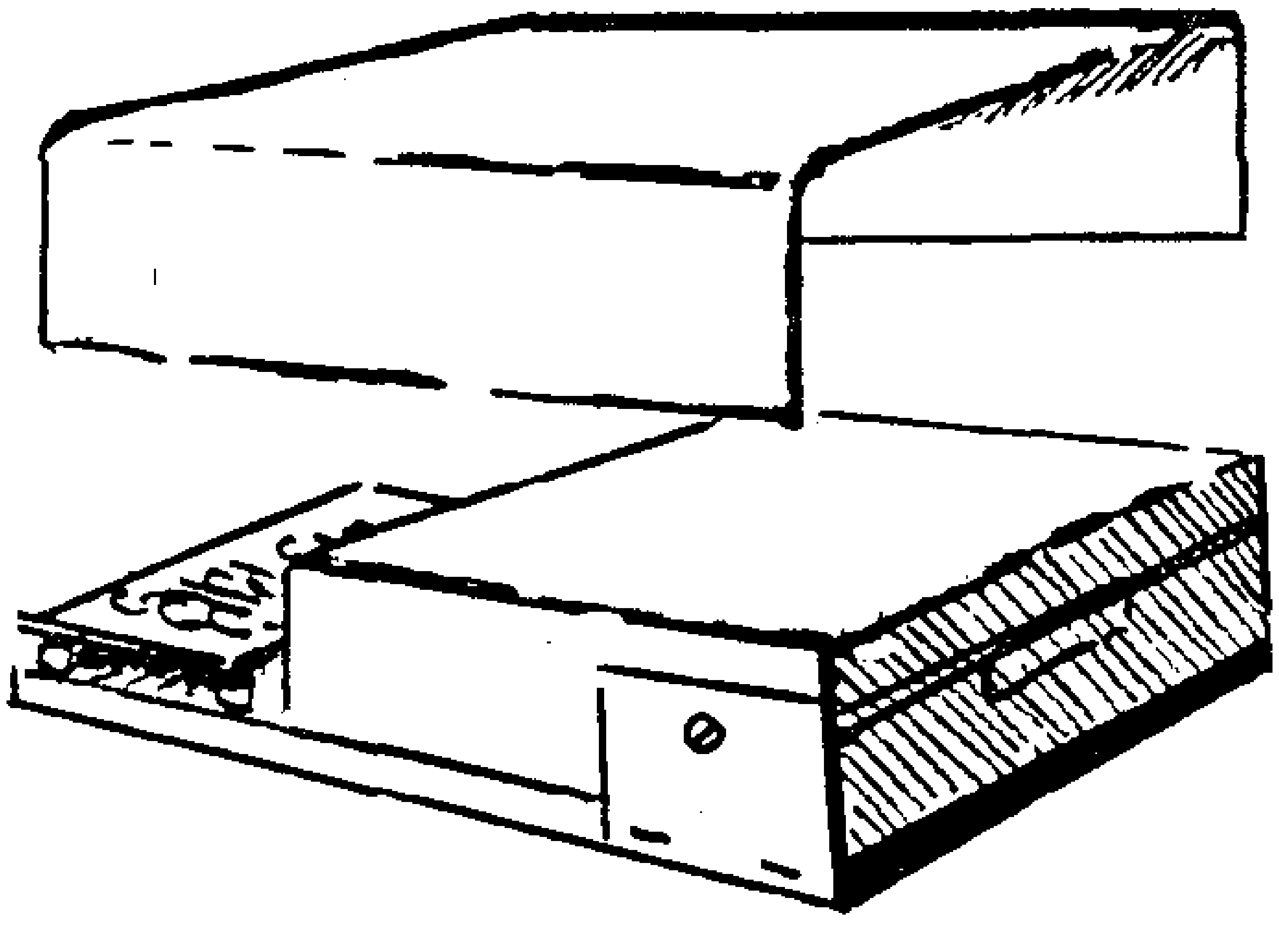
In the manual are detailed instructions on how to manipulate some actual code. There are 5 sample files to play with. I would recommend that these are followed through, before attempting to work on a program that you are writing. XB:BUG is NOT a beginners program, it is very powerful and as such has the capability to destroy a program in memory! If this were to happen and you saved the resultant memory to disk.....you may be cussing for a long time! So what GRADE do I give XB:BUG? I have to give it an A in everything except Ease of Use. Why? As I stated above, this program is not for the novice. If they feel they are buying a program that will teach them XB programming or literally do it all for them. They are sadly mistaken.

Did you know that...?

by Chick De Marti

KEN HAMAI NIGHT

At this month's meeting we enjoyed athe antics of Ken Hamai. He had a TI disk-drive for sale and to prove that it worked (also to show how inexpensive (" I'm talking CHEAP!") it can be to put together your own disk drive...HE BUILT ONE RIGHT THERE IN FRONT OF US! He started with a 1X8 piece of wood, with a notch cut out in the back for a recepticle to attach an A/C cord; dressed up the front of the wood with black masking tape, and NAILed a Radio Shack power supply to the back end of the board...(these were kept up inthe air with small pieces of hollow plastic tubeing). He then stapled a piece of cardboad to either side of the disk drive and screwed it to the side of the drive. " We don't want our drive to move around." he explained. He finished the job off by making a cover of " genuine Xerox Cardboard ", see drawind, and stapled it to the wood base. Then the moment of truth...will it work? Yes, it did, to the delight (and amazement) of all present. It was an entertaining and informative demo, and I look foward to his next 'performance'.



<*> <*> <*> <*> <*> <*> <*>

QUICKIE

From the Eugene 99/4A User Group come this idea:

If FCTN 4 is too much of a stretch, try; (hold down at the same time) FCTN J and spacebar.

" It works !"

!! SCOOP !!

Those of you who recieve the R.O.M. News- letter, read in the May issue that Ken (see 'HAMAI's hardware #3) is planning to somehow "move the Widget out of the way!". Well, it's done - I SAW IT!

At the L.A.99ers May meeting, he had...not one, but TWO widgets, hooked up in series, setting on the top of of console, just over the keyboard, and truely - OUT OF THE WAY!

A long extention strape would have satisfied most people, but Ken also added a short extention strape to connect his first Widget to one of the ports in the second Widget. Actually allowing 5 cartridges to be made available at the slip of two switches. MAH-VELOUS - simply MAH-VELOUS!

What's next, Ken???

<*> <*> <*> <*> <*> <*> <*>

THE OLD ONELINE
(by John Witte)

This month's "tinygram" comes from the Greater Omaha TI User Grope Newsletter. In XBasic it meets the requirement of being displayed on one screen only..."and not a bad little game to boot".

```
1 CALL CLEAR :: CALL COLOR(10
,16,7,2,11,11):: B,Q=0 :: W=I
NT(24*RND)+1 :: FOR T=1 TO 7
:: CALL VCHAR(5-(T>2)-(T=3),T
+12,42,VAL(SEG$("1353352",T,1
))):: NEXT T
2 DISPLAY AT(20,2):"GUESS?":"
#1=";R(1);"#2=";R(0)
3 FOR X=1 TO 6 :: FOR Y=15 TO
19 :: CALL SOUND(1,+5,0)
4 IF Q=0 THEN P=1+(P=1):: ACC
EPT AT(22-P,8)BEEP VALIDATE("
123456789"):Q :: B=B+Q
5 CALL HCHAR(6-X,Y,111):: IF
B>W THEN 7 ELSE Q=Q-1

6 NEXT Y :: NEXT X
7 DISPLAY AT(21-(P=1),8):"WIN
S!" :: R((P=1)+1)=R((P=1)+1)+
1 :: FOR J=5 TO 12 :: CALL SO
UND(600,440-11*J,0):: CALL HC
HAR(16-J,13,,111,7):: NEXT J
:: GOTO 1
```

(Did You Know ... cont.)

VEEP_WORKING_TO_STAMP_OUT_SEX

Beginning immediately, all words used in PCAG documents will be sexually neutral. This is in keeping with our policy of designating SIG leaders as Chairpersons and the Mailman as the Mailperson.

Inparticular, reference documents henceforth will be referred to as "person-uals" and archival storage will be called "its-tory files". Use of such terms is no longer optional, but person-datory!

A committee has been established to ferret out and neuter any additional terms that might slip into Bacon_Bits. The committee is headed by V.P. Karen Leeperson.

This important bit of information was gleened from the BIBMUG Newsletter (I was going to say "page 12"...but in days gone by, a PAGE was a young boy!)...sorry 'bout that.

<*> <*> <*> <*> <*> <*> <*>

SLASHED_ZERO

From "on the back burner" by Jean Wilcox (seen in the SUNCOAST BEEPER), if your printer doesn't have a slached zero in TI-writer, use this translit- ering command:

TL. 48:48,8,47

It prints your normal zero (CHR$(48)), then backspaces (CHR$(8)) and then prints a slash (CHR$(47)) over the existing zero. Slashing zeros in dates or zip codes for your letters serves no purpose whatsoever.

<*> <*> <*> <*> <*> <*> <*>

AN INTERESTING FORMULA"

```
10 PRINT "Enter your score a
and their score"
20 INPUT "(seperated by a co
mma)":YS,TS
30 W=W+ABS(YS>TS)
40 L=L+ABS(TS>YS)
50 PRINT "Wins=";
W,"Loses=";L
60 PRINT
65 CT=CT+1 :: IF CT=3 THEN END
70 GOTO 20
```

Line 65 is an afterthought.

By the way, for the complete version of Ken Hamai's 'How to' project, it is included else- where in this newsletter.

Chick

THE IRS AND "WORKING AT HOME"

I am semiretired and do some of my work at home...which poses the problem, "what about the IRS?"Rules regarding deductions of home office costs, depreciation of equiptment, etc. can be found in various IRS publications. None of them too clear except to warn you, "it ain't covered!" Some of the rules regarding home office expences states, "The portion of the home needs to be used exclusively on a regular basis for business purposes. The home office must be the 'principal place of business'. The use of the home office must be for the convenience of the employer". How many of those who bring home work, to help the boss, to catch, etc. fit all these rules? Also there are rules that office equiptment cannot be (usually) depreciated over a period of less then five years. Any person who owns a computer knows his gear is obsolete WHEN HE BUYS IT!

The IRS should reconoze the facts of life and allow some reasonible, fair breaks to thos of us working at home. I'll bet those portable computers the IRS bought for their staff will be long gone and replaced when that five-year peoiod is up. What is good for the goose should be good for the gander also.

How about another "DISPLAY AT" program that is written in BASIC? This one does it frontwards and backwards ... W O W !!!

```
10 CALL CLEAR
29 PRINT "Enter a sentence"
30 INPUT MSG$
40 INPUT "(F)oward or (B)ack
wards?":DIR$
50 L=LEN(MSG$)
60 IF DIR$="F" OR DIR$="f" T
HEN 140
70 J=(28-L)/2+L
80 FOR I=L TO 1 STEP-1
90 J=J-1
100 X=ASC(SEG$(MSG$,I,1))
110 CALL HCHAR(12,J,X,1)
120 NEXT I
130 GOTO 190
140 J=(28-L)-2
150 FOR COL=1 TO L
160 X=ASC(SEG$(MSG$,COL,1))
170 CALL HCHAR(12,J+COL,X,1)
180 NEXT COL
190 PRINT
200 INPUT "Another (Y/N) ":Y
210 IF Y$="Y" OR Y$="y" THE
N 10
220 END
```

Well...I'm out of coffee
see you next month ...

Chick

KIDS ############ Word Play April 1987

KORNER #####

The Game Room

by The TInman

```
100 REM FLIP
110 REM TI-99/4A EXTENDED BA
SIC
120 REM WESLEY R RICHARDSON
130 REM BLUEGRASS 99 COMPUTE
R SOCIETY
140 REM VARIABLES B,C(),C$,D
$,I,K,M,N,P,S
150 DIM C(9)
160 CALL CLEAR
170 DISPLAY AT(10,11):"F L I
 P"
180 DISPLAY AT(12,2):"BY WES
LEY R. RICHARDSON"
190 CALL CHAR(92,"FFFFFEFCFB
FEFEFEFEFEFEFEFBF8FFFFFFFF3F
3F3F3F3F3F3F3F3F0F0FFFFFF")
!ONE
200 CALL CHAR(96,"FFFFFCF8F1
F3FFFFFFFFFFCF8F0F0FFFFFFFF3F
1F8FCFCFCF8F1F3FFF0F0FFFFFF")
! TWO
210 CALL CHAR(100,"FFFFFCF0F
3FFFFFFFFFFFFF3F0F8FFFFFFFF3
F1F8FCFCF1F1FCFCF8F1F3FFFFF"
)! THREE
220 CALL CHAR(104,"FFFFF3F3F
3F3F3F3F0F0FFFFFFFFFFFFFFFFF
FFFFFFF3F3F0F0F3F3F3F3FFFFF"
)! FOUR
230 CALL CHAR(108,"FFFFF0F0F
3F3F3F0F0FFFFFFFF0F0FFFFFFFF0
F0FFFFFFF3F1F8FCF8F1F3FFFFF"
)! FIVE
240 CALL CHAR(112,"FFFFFCF8F
3F3F3F0F0F3F3F3F8FCFFFFFFFF3
F1FCFFFFF3F1FCFCFCF1F3FFFFF"
)! SIX
250 CALL CHAR(116,"FFFFF8F8F
FFFFFFFFFFFFEFEFCFCFFFFFFFF0
F0FCFCF9F9F3F3F7F7FFFFFFFFF"
)!SEVEN
260 CALL CHAR(120,"FFFFFCF8F
3F3F3F8F8F3F3F3F8FCFFFFFFFF3
F1FCFCFCF1F1FCFCFCF1F3FFFFF"
)! EIGHT
270 CALL CHAR(124,"FFFFFCF8F
3F3F3F8FCFFFFF3F8FCFFFFFFFF3
F1FCFCFCF0F0FCFCFCF1F3FFFFF"
)! NINE
280 CALL CHAR(128,"3F1F1C1C1
C1C1F1F1C1C1C1C1C1C3EF8F80
8000020E0E020000000000000000"
)! F
290 CALL CHAR(132,"3E1C1C1C1
C1C1C1C1C1C1C1C1C1F3F00000
000000000000000000008080F8F8"
)! L
300 CALL CHAR(136,"070101010
10101010101010101010107F0C0C
0C0C0C0C0C0C0C0C0C0C0C0C0F0"
)! I
310 CALL CHAR(140,"3F1F1C1C1
C1C1C1F1F1C1C1C1C1C1C3EE0F01
B0B0B0818F0E000000000000000"
)! P
320 REM INITIALIZE
330 B=99999
340 CALL CLEAR
350 CALL SCREEN(6)
360 CALL MAGNIFY(4)
370 RANDOMIZE
380 CALL CHARPAT(58,C$)
390 CALL CHAR(37,C$)
400 FOR I=1 TO 3
410 CALL CHARPAT(87+I,C$)
420 CALL CHAR(39+I,C$)
430 NEXT I
440 C$="FFFFFFFFFFFFFFFF"
450 C$="0000000000000000"
460 CALL CHAR(36,C$)
470 CALL CHAR(91,C$)
480 CALL CHAR(90,D$)
490 CALL COLOR(3,2,15)
500 CALL COLOR(4,2,15)
510 CALL COLOR(8,9,15)
520 REM RESTART POINT
530 N=-1
540 FOR I=1 TO 4
550 CALL SPRITE(#(I+9),124+4
*I,12,16,32+32*I)
560 NEXT I
570 FOR I=1 TO 13
580 CALL HCHAR(I+7,3,36,13)
590 NEXT I
600 DISPLAY AT(8,15):"GOAL Z
 ZZZ"
610 DISPLAY AT(9,22):"Z[Z"
620 DISPLAY AT(10,22):"ZZZ"
630 DISPLAY AT(12,15):"PRESS
Z [[["
640 DISPLAY AT(13,22):"[[["
650 DISPLAY AT(14,22):"[[["
660 DISPLAY AT(16,15):"RZRES
ET QZQUIT"
670 DISPLAY AT(18,15):"MOVES
Z     0"
680 DISPLAY AT(20,15):"BEST
Z"
690 GOSUB 1670
700 FOR I=1 TO 9
710 C(I)=9
720 CALL SPRITE(#I,88+4*I,9,
59+34*INT((I-1)/3),34*(I-3*I
NT((I-1)/3))-15)
730 NEXT I
740 P=1+INT(9*RND)
750 M=7
760 IF (P=1)+(P=3)+(P=7)+(P=
9)THEN M=10
770 IF (P=5)THEN M=11
780 GOSUB 1340
790 REM MAIN LOOP
900 GOSUB 1440
810 GOSUB 1700
920 ON K GOTO 830,890,940,10
00,1050,1120,1170,1230,1280
830 REM K=1
840 P=1 :: GOSUB 1340
850 P=2 :: GOSUB 1340
860 P=4 :: GOSUB 1340
870 P=5 :: GOSUB 1340
880 GOTO 800
990 REM K=2
900 P=1 :: GOSUB 1340
910 P=2 :: GOSUB 1340
920 P=3 :: GOSUB 1340
930 GOTO 800
940 REM K=3
950 P=2 :: GOSUB 1340
960 P=3 :: GOSUB 1340
970 P=5 :: GOSUB 1340
980 P=6 :: GOSUB 1340
990 GOTO 800
1000 REM K=4
1010 P=1 :: GOSUB 1340
1020 P=4 :: GOSUB 1340
1030 P=7 :: GOSUB 1340
1040 GOTO 800
1050 REM K=5
1060 P=2 :: GOSUB 1340
1070 P=4 :: GOSUB 1340
1080 P=5 :: GOSUB 1340
1090 P=6 :: GOSUB 1340
1100 P=8 :: GOSUB 1340
1110 GOTO 800
1120 REM K=6
1130 P=3 :: GOSUB 1340
1140 P=6 :: GOSUB 1340
1150 P=9 :: GOSUB 1340
1160 GOTO 800
1170 REM K=7
1180 P=4 :: GOSUB 1340
1190 P=5 :: GOSUB 1340
1200 P=7 :: GOSUB 1340
1210 P=8 :: GOSUB 1340
1220 GOTO 800
1230 REM K=8
1240 P=7 :: GOSUB 1340
1250 P=8 :: GOSUB 1340
1260 P=9 :: GOSUB 1340
1270 GOTO 800
1280 REM K=9
1290 P=5 :: GOSUB 1340
1300 P=6 :: GOSUB 1340
1310 P=8 :: GOSUB 1340
1320 P=9 :: GOSUB 1340
1330 GOTO 800
1340 REM SET COLOR
1350 IF C(P)<>9 THEN 1400
1360 C(P)=15
1370 CALL COLOR(#P,15)
1380 CALL HCHAR(12+INT((P-1)
/3),23+P-3*INT((P-1)/3),48+P
)
1390 RETURN
1400 C(P)=9
1410 CALL COLOR(#P,9)
1420 CALL HCHAR(12+INT((P-1)
/3),23+P-3*INT((P-1)/3),91)
1430 RETURN
1440 REM CHECK FOR SOLUTION
1450 N=N+1
1460 DISPLAY AT(18,26-LEN(ST
R$(N))):N
1470 IF C(5)<>9 THEN 1660
1480 FOR I=1 TO 4
1490 IF C(I)<>15 THEN 1660
1500 NEXT I
1510 FOR I=6 TO 9
1520 IF C(I)<>15 THEN 1660
1530 NEXT I
1540 B=MIN(B,N)
1550 GOSUB 1670
1560 DISPLAY AT(22,3):"SOLUT
ION !!"
1570 DISPLAY AT(23,3):"ONL)"
;N-M;"E(TRA MOVES"
1580 CALL SOUND(500,440,0)
1590 CALL KEY(0,K,S)
1600 IF S=0 THEN 1590
1610 IF K=81 THEN 1790
1620 IF K<>82 THEN 1590
1630 DISPLAY AT(22,1):""
1640 DISPLAY AT(23,1):""
1650 GOTO 520
1660 RETURN
1670 REM BEST SCORE
1680 DISPLAY AT(20,22):SEG$(
"     ",1,5-LEN(STR$(B)));ST
R$(B)
1690 RETURN
1700 REM WAIT FOR KEY
1710 CALL KEY(0,K,S)
1720 IF S=0 THEN 1710
1730 IF K=81 THEN 1790
1740 IF K=82 THEN 520
1750 IF (K<49)+(K>57)THEN 17
10
1760 K=K-48
1770 IF C(K)=9 THEN 1710
1780 RETURN
1790 REM QUIT
1800 CALL CHARSET
1810 END
```

FLIP FLOP

This little game of "Flip-Flop" will test your patience. When you run the program you will be presented with a square divided into nine small connected squares, one or more of which will be white and the rest red. Your job is to change the colors so that finally you have the center square red and all the rest white. You do this by pressing the number of any white square and the ajacent colors will flip from white to red or vice-versa. There is a scoring area that keeps track of the number of times you hit the keys and another area that records the shortest time for a player. I did it once in 17 tries but have not come anywhere near that on subsequent games.

HAMAI'S hard WARE #5

WARNINGWARNING***

This project requires some skills and knowledge in electronics assembly. Incorrect assembly could result in burning up your disk drive. If you are not sure how to connect the parts, contact me or somebody who can help. In any event, neither I or the ROM will cover you for any damages or losses resulting from the use of this power supply as suggested by this article and you are using it solely at your own risk.

This power supply was only tested with the Shugart 400, 400L and 450 disk drives. These drives have a low power requirement and as a result work well with this supply. Disk drives which require stronger power supplies will not work.

PARTS LIST

1. Radio Shack 277-1016 power supply chassis
2. Radio Shack 273-1511 12.6volt, 3amp transformer
3. 1/2 amp fuse and holder
4. SPST bat switch
5. 5 ft. of lamp cord and plug
6. Three 6 inch lengths of 20-22ga stranded wire, different colors
7. Male plug for disk drive power connector
8. Small piece of heat shrink tubing 1/16th inch size

All of the above items except for item 7 are available at Radio Shack stores. The disk drive connector plug is available from R&D Electronic Supply, 100 E. Orangethrope Ave. Anaheim, CA 92801.

If you plan to use the same thing I used on the demo, then you will need to have a TI computer transformer. If you have the TI transformer then you would not need to purchase items 2, 3, 4, and 5 on the parts list.

When you purchase the power supply board, you will note that it comes with instructions on a suggested wiring scheme. These instructions also recommend the use of an 18volt transformer and 2amp fuse. The reason I *recommend the other transformer is* because the power supply does not have to work so hard to regulate the output voltages and the lower amperage fuse gives quicker response to an overload. An added plus is that the 12.6volt transformer costs less.

Step 1. - See fig. 1 for suggested hookup for the transformer and the power supply. You will note this is the same as shown in the Radio Shack diagram. I have included some notes for clarity.

Step 2. - Double check your connections and then plug in your supply to an outlet to test it out. Be sure to turn on the power supply switch located on the board (see fig. 3). Using a suitable volt meter and fig. 2, check that you get the indicated output voltages when you test the +5 and +12 pins and ground. The voltages MUST be *pretty close. DO NOT use the power* supply if you find it is off by 1/2 volt or more, especially on the +5 volt pin. If the voltages are way off, I suggest you return the board and get another one.

Step 3. - Disconnect the power to the supply and carefully bend the -5 volt pin out of the way or cut it completely off. Then solder one of the 6 inch lengths of 20-22ga wire to each of the remaining pins. Use a piece of heat shrink tubing over each soldered connection for insulation. Use yours or a friend's blow dryer at the High setting to shrink the tubing.

Step 4. - Referring to fig. 4, assemble the three wires you solderuo to the pins into the power connector for the disk drive. Double check your wiring and test the connector with your *voltmeter to be sure that you have the* wires in the correct socket positions.

That's all there is to the wiring. If you connected up your disk drive now, it should work.

One more thing. I have not included plans for a cabinet for the disk drive and power supply. You will need to build one to hold your components together. For my demo model, I used the TI computer power supply transformer and eliminated the extra transformer, fuse, switch and lamp cord. And mounted the disk drive to a piece of plywood and covered the whole thing with a cardboard to keep the fingers and dust out. I suggest you take your own measurements for the design of the cabinet. See fig. 5 for a suggested configuration.

Using the above and one of the TI/Shugart 400L disk drives you should be able to set up a second or third disk drive drive for your system AND this power supply for less than $40.00, including the cardboard and nails.

Bye for now. Be especially careful out there and tell them all you saw it FIRST!...in the ROM...Ken H.

THE R O M NEWSLETTER
USERS GROUP OF ORANGE COUNTY
17301 SANTA ISABEL STREET
FOUNTAIN VALLEY, CA 92708

This article was originally published in the September 1985 issue.

DISK DRIVE POWER SUPPLY

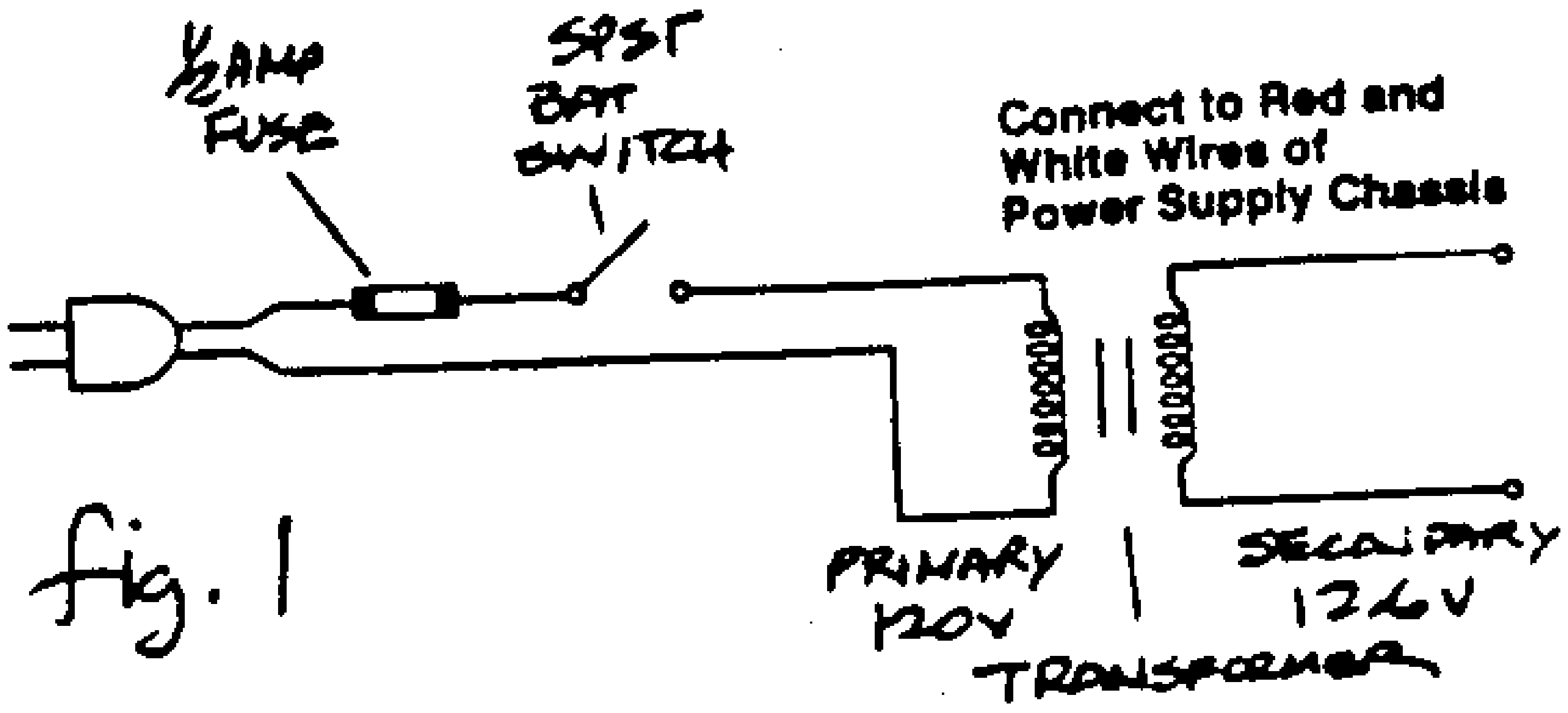
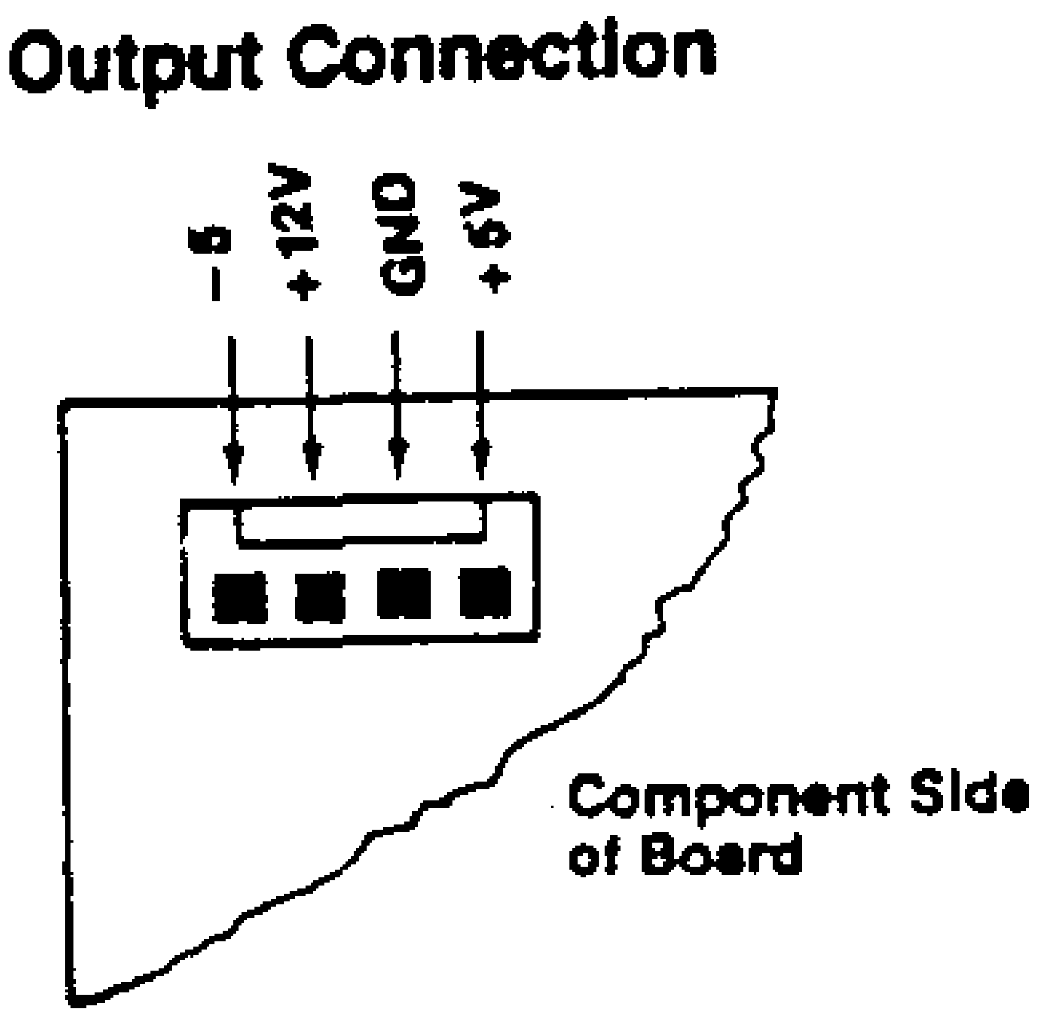


fig. 1



−5V NOT USED

fig. 2

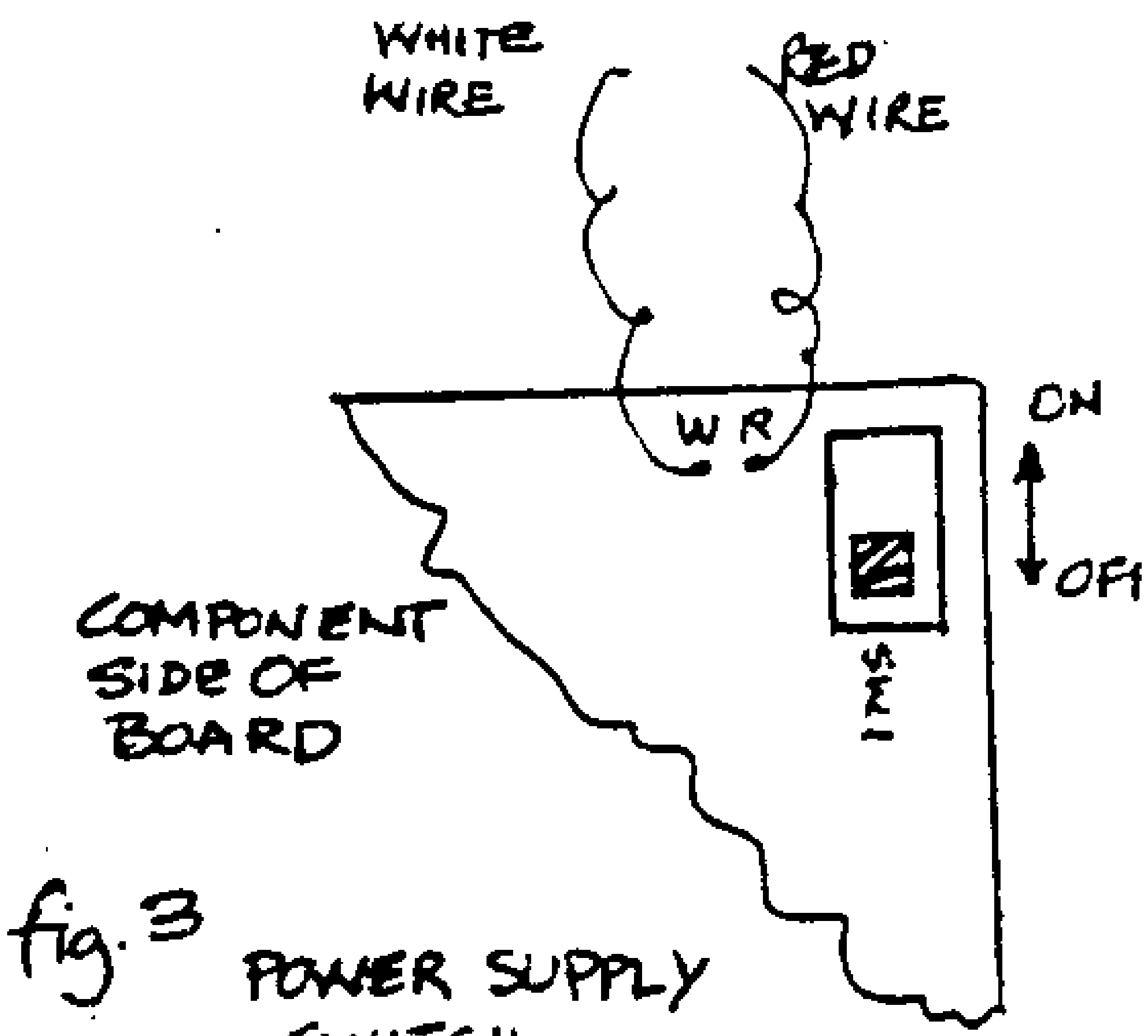
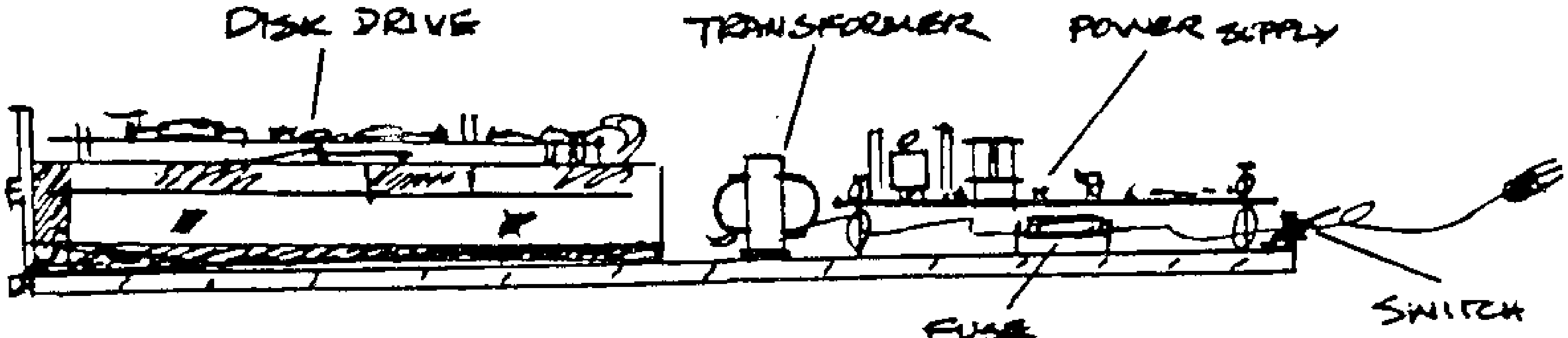


fig. 3 POWER SUPPLY SWITCH



POSSIBLE MOUNTING CONFIGURATION

fig. 5

QB MONITOR ~ QB-99'er NEWSLETTER

4A/TALK

DELETE FILE	TE-II					HELP				
PRINT ON	PRINT OFF	HALF/FULL DUP	BUFF ON/OFF	SAV/CLR BUFF	SEND ASCII	CATALOG	CONFIGURE	XMODEM XFER	AUTO-DIALER	QUIT

CASI:ICON

MODEM BAUD	PRNT ON/OFF	MODEM PARITY	MODEM PORT	PRINT PARITY	PRINT PORT	PRINT BAUD			SCR'N WIDTH	QUIT
N,=SND FILE	B=DUMP BUFF	Y=CLEAR LOG	CANCEL	WINDOW RIGHT	K=SET TIMER	TEXT COLOR	SCREEN COLOR	,=LOG ON/OFF	FREEZE TOGL	QUIT

F-TERM

TOGGL PRNTR	LOAD BUFFER	XMIT NXT LINE	XMT X/ONOFF	RESET BUFFER	DUMP BUFFER	RESTRT PROG			QUIT	
E=TOGL ECHO						SCR'N COLOR				

TE-II

SPEAK	OUTPUT	CANCEL	TRANS	WRAP	CASE	PAGE			EXIT	
										QUIT

BASIC

DELETE	INSERT	ERASE	CLEAR	BEGIN	PROC'D	AID	REDO	BACK		QUIT

TI-WRITER

OOPS!	REFORMAT	SCREEN COLOR	NEXT PARA	DUPE LINE	LAST PARA	WORD TAB	NEW PARA	NEW PAGE	WORD WRAP	
DELETE CHAR	INSERT CHAR	DELETE LINE	ROLL DOWN	NEXT WINDOW	ROLL UP	TAB	INSERT LINE	COMMAND/ESC	LINE #'S	QUIT

MULTIPLAN

HOME	TAB	NEXT UNL CELL	FORW'D CHAR	FORW'D WORD	CHNG WINDOW	REL/ABS REF				CANCEL
LOWER RIGHT			BACK CHAR	BACK WORD		HELP	RECALC	BACKSPACE	DEL FORWARD	

RAVE_KYBD

OOPS	REFORMAT	SCREEN COLOR	NEXT PARA	DUPE LINE	LAST PARA	WORD TAB	NEW PARA	NEW PAGE	WORD WRAP	LINE #'S
LOWER RIGHT	REL/ABS REF	FORWARD CHAR	BACK CHAR	BACK WORD	FORWARD WORD	HELP	RECALCULATE	NEXT WINDOW	NXT UNL CEL	RAVE 99/101

EDIT/ASM

DELETE CHAR	INSERT CHAR	DELETE LINE	ROLL UP	NEXT SCREEN	ROLL DOWN	TAB	INSERT LINE	ESCAPE		QUIT
DELETE	INSERT	ERASE	CLEAR	BEGIN	PROCEED	AID	REDO	BACK		QUIT

FORTH

							INSERT LINE			FORTH
DELETE	INSERT	ERASE	NEXT SCREEN	NEXT WINDOW	LAST SCREEN	DEL)ENDOLIN	COPY LINE	EXIT EDITOR		

DISKO

			FCTN E=UP	FCTN X=DOWN	FCTN S=LEFT	FCTN D=RGHT				DISKO
HEX DISPLAY	ASCII DISPL	LEAVE PROGRAM	BACK 1 SCTR	RSTRT SUBPRG	FRWRD 1 SCTR		REWRITE SCTR	RSTRT PROGRM		QUIT

ADV/DIAG

E=PG/BUF UP	X=PG/BUF DN									ADV DIAGNOS
DELETE	INSERT	ERASE	INIT SCTR/TK	EXIT PROGRM	PREV SCTR/TK	HELP	RETES BADMEM	ESCAPE	SCREEN DUMP	ASCII/HEX

GRAPHX

										GRAPHX
SLOWER	FASTER	DRAW	ERASE	NO HELP	ZOOM	COLORS	LINES	CIRCLES	COPY	MENU

CHESS

NO CONTROL	KEY PRESS	REQUIRED		1=MODE CHANG		(=HELP				CHESS
WITH FUNCTN	D=OFFR DRAW	3=TIME OUT	4=ERASE	E=RESIGN	P=ARANG POSN	S=SWITCH	8=REPLAY	I=BACK UP	X=SAVE GAME	*=QUIT

CREDITS

COMPILED BY		ED MACHONIS								
PRINTED	USING	"PRINTSTRIP"	WRITTEN BY	MIKE	MACHONIS	AS	PUBLISHED IN	MICROPENDIUM	AUGUST 1986	

ASSEMBLY DISK CATALOG, REVISION FOR HARD DISK

by Tom Freeman

In April, 1987 I published source code for an assembly language catalog routine that worked on interrupts. At that time I promised a version for hard disk controllers - and here it is! In order not to bore you too much, I have changed quite a number of the routines, in addition to adding the code for the hard disk. There are a number of new "bells and whistles," and I have tried to indicate those lines which are changed and which are new, so that if you wish you may add the code for the bells without the hard disk code.

The following features have been added: **1)** scrolling up and down line by line as well as by page, using the FCTN X and FCTN E keys, (but the scroll will not go beyond the end or the beginning, whereas the page scroll is circular) **2)** auto repeat if you hold down one the four active keys for about one second (but the scroll is much faster for lines, 3/6Ø sec rather than 24/6Ø sec for pages, so you can stop in time) **3)** only as much VDP memory is cleared as is needed for the number of pages used, leaving less chance of wiping out strings (in XB) or program (in E/A Basic) **4)** if you forget to add the period at the end of the device name, the program will add it for you, **5)** you may enter just a drive number and the program will convert it to "DSKx." for you, and **6)** a running count is kept of the number of files (plus subdirectories in the case of a hard disk) near the top of the screen.

This program makes use of the CFI (convert floating point to interger) routine in the console rather than the CSN (convert string to number) as I felt it worked a little faster. I use my own routine to convert to ASCII.

Some additional bits of information. An error crept into the 3rd paragraph of the previous article - **all** 8 bytes of the floating point number are significant. Unused places must be >ØØ. However **ZERO** itself is represented by >ØØ,>ØØ and the other 6 bytes are undefined. Also please note that the code for the Hard Disk directory is based on MYARC's present Personality Card, which lists subdirectories first, and then files beginning in "record" 115. There is a rumor that this may be switched in their new combined floppy/hard disk controller when it is released. If so it should be simple to make the switch of the two text lines representing files and subdirectories, and if the record number is changed, that is also easily modified.

Good luck, sorry it is so long - writing compact code is not yet my strength!

```
* New lines are marked with an "*" at the end of the line
* Changed lines with a "%",some only text is changed or
* label changed or added.
* I have not included GPLLNK, DSRLNK, and VDP utilities
* as they are unchanged EXCEPT for the error at label
* WTR where it should read MOV   *13,12
* Sorry about my laziness in not using R for registers!
       DEF  START,OFF,ON
PABLOC EQU  >ØFØØ
FAC    EQU  >834A
* AR6    EQU  >835C      This line is deleted          %
STATUS EQU  >837C
KEY    EQU  >8375
GPLWS  EQU  >83EØ
VBF1   EQU  >ØF4Ø                                      %
* VBF2   EQU  >1Ø8Ø      Line deleted                  %
VBF3   EQU  >1ØØØ                                      %
VBF4   EQU  >14ØØ                                      %
***************************************************
OFF    CLR  @>83C4       Clear ISR hook-turn off       *
       RT                interrupts and return         *
ON     LI   Ø,START      Load start of routine in ISR
       MOV  Ø,@>83C4
       RT
START  CB   @KEY,@CTRLC  Is CTRL C pressed?
       JEQ  S2           Yes, begin
       RT                No, return
S2     LWPI >83CØ        Interrupt workspace
       MOV  13,@SAV13    R13-15 nced to be saved for rtn
       MOV  14,@SAV14    -destroyed by CFI routine below*
       MOV  15.@SAV15
```

```
       CLR  @>83C4       Turn off interrupts temporarily
       LWPI WS           Our Workspace
* The next 8 lines are needed in Basic only, because all
* text must have basic bias of >6Ø added
       ABS  @BIASCK      Have we modified text already?
       JNE  S1A          Yes, skip next part
       SETO @BIASCK      Mark the change
       LI   Ø,DEV        Beginning of text to be changed
       LI   2,TYPES-DEV  Length (end-beginning)
S1     AB   @BIAS,*Ø+    Add the bias one byte at a time
       DEC  2            More?
       JNE  S1           Yes, go back
       LI   Ø,4          2nd XMLLNK table address is    *
       A    @>ØCFC,Ø     specified at >ØCFC. CFI(Convert*
       MOV  *Ø,@CFI      Floating Point to Integer) rou-*
*                        tine is 3rd entry,4 bytes from *
*                        the beginning.                 *
S1A    LI   Ø,>Ø2Ø4      Change scrn image table from Ø %
       BLWP @VWTR        to >1ØØØ(>4ØØ*4)-saves orig scr%
       LI   Ø,>Ø1FØ      Text mode
       BLWP @VWTR
*A copy of value in VDP REG 1 (now in LSB of RØ) must be
*placed @>83D4 because the value there is transferred to
*VDP REG 1 at every key press
       MOVB @WS+1,@>83D4
       LI   Ø,>Ø717      1=FG color,7=BG change if you
       BLWP @VWTR                                  like
S3     LI   Ø,>ØØ5Ø      WRITE address for VDP >1ØØØ    %
       LI   2,>8CØ2      VDPWA                  reversed
       MOVB Ø,*2         Move LSB of >5ØØØ first        %
       SWPB Ø
```

```
      MOVB Ø,*2          Now MSB
      DECT 2             VDPWD (>8CØØ) - as each byte is
*                        moved here, the address at >8CØ2
*                        auto-increments - Handy!
* Following line changed because in E/A Basic a program *
* might reside at the top of VDP                         *
      LI   1,96Ø         clear screen                    %
      LI   Ø,>8ØØØ       space with basic bias,use >2ØØØ
*                        if not in basic
S4    MOVB Ø,*2          Because of auto-increment each
      DEC  1             byte written goes to next, with-
      JNE  S4            out changing R2
      LI   Ø,>1Ø2A       3rd Row, Col. 1Ø                %
      LI   1,DEV         Text
      LI   2,9           Remember new screen image table
      BLWP @VMBW         write on screen
      AI   Ø,9           Prepare for input
      CLR  2             Counter
      LI   3,DEVBUF+1    For storage                     %
S5    BLWP @KSCAN        Look for key press
      MOVB @STATUS,1     Key pressed?
      JEQ  S5            No, go back
      MOVB @KEY,1        Yes put value in MSB of R1
      CB   1,@ENTER      Enter Key?
      JEQ  S6            Yes, process
      MOVB 1,*3+         Store value,increase buffer pos.
      AB   @BIAS,1       Add Basic Bias to R1
      INC  Ø             Next position
      BLWP @VSBW         Write on screen
      INC  2             Increase the counter
      JMP  S5            Go for more
S6    MOV  2,2           Enter pressed without text?
      JNE  S6B           No, go on                       %
      B    @ENDEND       Yes, branch to end
* The next 13 lines allow for input of a number only     *
* i.e. length of 1.  Number X is then replaced by        *
* DSKX.  You can delete them, but replace S6B with S7    *
* two lines above                                        *
S6B   CI   2,1           Length = 1?                     *
      JNE  S7            No, go on                       *
      MOVB @DEVBUF+1,@DEVBUF+4 Move # to 4th place       *
      LI   Ø,DEVBUF      Now move length byte and DSK    *
      LI   1,DSK1        to DEVBUF area                  *
      LI   2,4           4 to do                         *
S6A   MOVB *1+,*Ø+       move, advance source & destin.  *
      DEC  2             More?                           *
      JNE  S6A           Yes go back                     *
      INC  Ø             Advance counter           .     *
      INC  1                "     "                      *
      MOVB *1,*Ø         Now move the period             *
      JMP  S7B           And continue processing         *
S7    BLWP @VSBR         Read the last character
      CB   1,@PERIOD     Is it a period?
      JEQ  S7A           Yes, go on                      %
      MOVB @DSK1+5,*3    No, better add it!!!            *
      INC  2             Increase counter                *
S7A   SWPB 2             Count in MSB of R2
      MOVB 2,@DEVBUF     Len byte at start of storage    %
S7B   LI   Ø,VBF3        This is the screen image table  %
      LI   1,TIT1        Text
      LI   2,12Ø         3 lines
* Because a new area, DEVBUF has been added to the 1st   *
* 9 bytes of DSKPAB, if the len byte is 1st in DEVBUF    *
* the whole PAB can be written at once                   *
      BLWP @VMBW         Write
```

```
      LI   Ø,PABLOC      Open mode
      LI   1,DSKPAB
      LI   2,48                                          %
      BLWP @VMBW         Write whole PAB                 %
* Thus next 6 lines were deleted                         *
      MOV  @PABPT,@>8356 Pointer to location of len byte
      BLWP @DSRLNK       Open the file
      DATA 8
      JEQ  S3            Error, go back
      CLR  @HARD         Will be the flag for hardisk    *
      BL   @RECRD        "Read" 1st record (will contain
*                        disk name,then 3 #'s (Ø,total   %
*                        sectors, number available)      *
      BL   @CLRBUF       Clear buffer space
      LI   1,PROBUF      Where string will go
      BLWP @VMBR         Read string into it-see RECRD to
*                        see what RØ,R2 have become
      A    2,Ø           Next item is a number(fltng pt)
* The next 19 lines beginning with AI 2,Ø, from the ori-
* ginal program are deleted, and replaced with the code
* ending at label S1Ø
* The 6 lines beginning with CI 3,>CØØ may be deleted    *
* if you do not wish Hard Disk cagalog option            *
      AI   Ø,1Ø          Advances past 1st #,which is Ø  *
*                        in 1st record,and skips past    *
*                        length of next FlPt #           *
      BL   @GETNUM       Places integer in R3            *
      MOV  3,@TOTSEC     Saves it                        *
      CI   3,>CØØ        >B4Ø=288Ø,max sectors on a quad*
* >CØØ=3Ø72 sectors=786K bytes,less than 1 MEG=not hard  *
      JLE  S8            If greater,then Hardisk         *
      SETO @HARD         Flag it                         *
S8    CI   3,2           Hardisk subdirectories yield Ø  *
      JGT  S9            or 2,thus others are floppies   *
      SETO @HARD         Mark it and skip,as there is no*
      JMP  S1Ø           useful info in rest of record   *
S9    AI   Ø,9           Next #,past length byte         *
      BL   @GETNUM       Get it in R3 as word            *
      MOV  3,@AVLSEC     Save it                         *
      S    @AVLSEC,@TOTSEC Now TOTSEC  is actually USED*
      MOV  @AVLSEC,1     Number to use for next routine  *
      LI   4,PROBUF+15   Where to put it                 *
      BL   @MOVNUM       Make it an ASCII decimal #      *
      MOV  @TOTSEC,1     Next number                     *
      LI   4,PROBUF+3Ø   Where                           *
      BL   @MOVNUM       ASCII decimal                   *
* Delete next line (and move label down)if not in Basic  *
S1Ø   BL   @ADD6Ø        Add Basic Bias to text          %
      LI   Ø,VBF3+9      Next three BLWP @VMBW instruc-  %
      LI   1,PROBUF      tions place the DISKNAME, AVAIL
      LI   2,1Ø          and USED in proper locations on
      BLWP @VMBW         screen
      AI   Ø,15                                          %
      AI   1,15                                          %
      LI   2,5                                           %
      BLWP @VMBW
      AI   Ø,1Ø                                          %
      AI   1,15
      BLWP @VMBW
      LI   9,VBF4         Initialize buffer to hold files
      CLR  @TOTAL         Line counter                   %
* Next line used for on screen file counter              *
      CLR  @COUNT         File,subdirectory counter      *
* Next 8 lines used for Hard Disk, can be deleted        *
      ABS  @HARD          Is this a hardisk?             *
```

```
       JEQ  GETPRO        No,go on                          *
       LI   Ø,VBF4        These 4 lines place the "title"*
       LI   1,TIT4        SUBDIRECTORIES on the first       *
       LI   2,4Ø          line                              *
       BLWP @VMBW                                           *
       AI   9,4Ø          Advance to next line              *
       INC  @TOTAL        And mark it                       *
GETPRO BL   @CLRBUF       Clear PROBUF
* Each "record" will now produce a string which is the
* Filename, then 3 floating point numbers:1) file type
* negative if protected,2)size in sectors 3)record length
* if not "program." Hardisk subdirectories always yield %
* 2, but this will be checked for.  Some DSR's place    *
* total number of bytes in 3rd #, if "program."  This   *
* number WILL be used.                                  *

CPØ    BL   @RECRD        [Note added LABEL]                %
       JEQ  END1          Null string=no more,jump to end
       LI   1,PROBUF+3    Read the name into PROBUF
       BLWP @VMBR
       A    2,Ø           Get to 1st number
       CLR  1
* 1st number is -5 to +6,so 1st byte is ALWAYS >4Ø      %
       INCT Ø             therefore 2nd byte is IT
       BLWP @VSBR
       LI   3,>592Ø       "Y" or " "
       SRA  1,8           Number in LSB,but sign bit there
       JLT  GP1           If negative, leave R3 alone
       SWPB 3             Put " " in MSB of R3
GP1    ABS  1             Now get the positive # 1-5
       MOV  1,8           Save R1
       MOVB 3,@PROBUF+33 Put "y" or " " in proper loc.  %
       SLA  1,3           multiply by 8
       AI   1,TYPES-8     Index to file type
       MOV  1,3           Use for next BL                   %
       LI   4,PROBUF+19
       LI   5,8                                             %
       BL   @MOV34        Move the TYPE to PROBUF
       AI   Ø,8           Next number(RØ already INC'd)     %
       MOV  Ø,7           Save it                           *
* Delete next line (LI 4,...)                           *
       BL   @GETNUM       Get number into R3                %
       MOV  3,1           Use in next routine               *
       LI   4,PROBUF+13   Where                             *
       BL   @MOVNUM       Place it there                    *
       CI   8,6           6=subdirectory-no useful info     %
       JEQ  GP2           So skip record length             %
       MOV  7,Ø           Restore RØ                        *
       AI   Ø,9           Next number
* Delete next line (LI 4,...)                           *
       BL   @GETNUM       Get number into R3                *
       MOV  3,1           Use in next routine               *
       JEQ  GP2           If zero don't use                 *
       LI   4,PROBUF+27   Where                             *
       BL   @MOVNUM       Place it there                    *
* Delete next line (and move label down)if not in Basic *
GP2    BL   @ADD6Ø        Now add Basic Bias to all
       MOV  9,Ø           Location in VDP buffer
       LI   1,PROBUF
       LI   2,4Ø
       BLWP @VMBW         Write it to the buffer
       A    2,9           Next position in buffer
       INC  @COUNT        File,subdirectory counter         *
       MOV  @COUNT,1      Use for next routine              *
       LI   4,PROBUF      Where                             *
```

```
       BL   @MOVNUM       Put the ASCII number there        *
       LI   3,3           3 places                          *
       LI   4,PROBUF+2    Not first 2                       *
       MOV  4,1           Save for VMBW                     *
       MOV  3,2                  .                          *
GP3    AB   @BIAS,*4+     Add the basic bias                *
       DEC  3             More?                             *
       JNE  GP3           Yes                               *
       LI   Ø,>1Ø73       Where on screen                   *
       BLWP @VMBW         Write it                          *
       INC  @TOTAL        Counter for lines                 %
       JMP  GETPRO        Back for more
* From here to END1A can be deleted if no Hard Disk     *
END1   ABS  @HARD         Hardisk?                          *
       JEQ  END1A         No,skip                           *
       CLR  @HARD         Yes,now clear,& we don't repeat*
       LI   Ø,PABLOC+7    Location of the RECORD NUMBER  *
       LI   1,>73ØØ       Record No. 115           (LSB)    *
       BLWP @VSBW         Change the PAB                    *
       MOV  9,Ø           Where we are in buffer            *
       LI   1,TIT5        "FILES"                           *
       LI   2,4Ø                                            *
       BLWP @VMBW         Write it to buffer                *
       INC  @TOTAL        Indicate next line                *
       AI   9,4Ø          and change buffer location        *
       JMP  GPØ           And go back for the files         *
END1A  LI   Ø,PABLOC      Note change of label              %
       LI   1,>Ø1ØØ       OPcode for close
       BLWP @VSBW
       MOV  @PABPT,@>8356
       BLWP @DSRLNK       Close the file
       DATA 8
       LI   4,21          Divide one R (4) into a contig-
       CLR  5             uous 2 word area(R5-6),integer
       MOV  @TOTAL,6      result in R5,remainder in R6
       DIV  4,5           R5=# of FULL pages of 21 files %
       S    6,@TOTAL      Top line,last page(full/empty) *
* TOTAL=actual # lines,but we'll start counter at Ø     *
       MOV  6,6           Anything on last page?            *
       JEQ  END2A         Skip next part, full page         *
       A    4,@TOTAL      TOTAL=top line 1st unused page *
END2B  S    6,4           # lines left on last page         *
       LI   3,4Ø          # bytes per line                  *
       MPY  3,4           Now R5=bytes left on page         *
*                         See explanation below of MPY      *
       AI   9,>4ØØØ       WRITE address of present R9       *
       LI   2,>8CØ2       VDPWD                             *
       SWPB 9             Move LSB first                    *
       MOVB 9,*2                                            *
       SWPB 9             Now MSB                           *
       MOVB 9,*2                                            *
       DECT 2             Now R2=VDPWD                      *
       LI   1,>8ØØØ       space w/ basic bias(>2Ø if not)*
END2C  MOVB 1,*2          Write one byte,address auto-inc*
       DEC  5             # bytes left to do                *
       JNE  END2C         More                              *
END2A  MOV  @TOTAL,7      TOP line 1st unused page          *
       MOV  7,4                                             *
       AI   4,-21         Top line of last page             *
       CLR  6             Initialize to first line          %
* The 8 lines after CLR 6 were changed,added to SCRO    *
       CLR  12            Auto repeat loop flag             *
END2   BL   @SCRO         Write to screen                   %
END3   BLWP @KSCAN        Look for key press
* Essentially everything from here to ENDEND is new, if *
* you don't wish auto repeat & scrolling leave the old  *
* routine alone, & take out apprcpriate part of SCRO    *
       LIMI 2          Enable interrupts for counter   *
       LIMI Ø                                          *
* The Equal bit at STATUS is set only if a NEW key is   *
* pressed since last call to KSCAN                      *
       MOVB @STATUS,1
       JNE  END3A       A new key                       %
       MOVB @KEY,1      Move value of key press to R1
       CB   1,@FF       This means NO key is presed     *
       JEQ  END3        So go back                      *
       ABS  12          Auto repeat loop started?       *
       JNE  END3B       Yes, go to processing           *
       MOVB @>8379,1Ø   VDP counter in MSB of R1Ø       *
       CI   1Ø,>4ØØØ    Is it less than >4Ø?            *
       JLT  END3        Yes, go back to KSCAN           *
       SETO 12          No, set the flag                *
       JMP  END3B       And now go on                   *
END3A  CLR  12          NEW key press went here,clear   *
       MOVB @D1Ø,@>8379 auto repeat flag,& VDP counter  *
* I changed what X and E do to be more consistent with  *
* other programs (X pages DOWN, i.e. HIGHER number)     *
END3B  MOVB @KEY,1      Get value of key into R1 again  %
       CB   1,@ENTER    Is it "enter"?
       JEQ  ENDEND      Yes, jump to end
       CB   1,@X        Is it "X"?
       JNE  END4        No, jump ahead
       AI   6,21        Add 21 lines                    *
       C    6,7         Tcp line 1st unused page?       *
       JEQ  END3C       Yes, go to reset R6             *
       C    6,4         Before tcp of last page?        *
       JLT  END3D       Yes, go on                      *
       MOV  4,6         No, SET R6 to tcp last page     *
       JMP  END3D       Go on (R6 never >top last page  *
END3C  CLR  6           Reset line counter to zero      *
END3D  ABS  12          In an auto repeat loop?         *
       JEQ  END2        No, go back to write            *
       LI   12,>18ØØ    Yes set counter to 24 for pages*
       JMP  END2        And go back to write            *
END4   CB   1,@FCTNX    Is it "FCTN X"?                 *
       JNE  END5        No, go on                       *
       C    6,4         Line counter at top last page?  *
       JEQ  END3        Yes,just go back to KSCAN       *
       INC  6           No, increase line counter       *
END4A  ABS  12          Auto repeat?                    *
       JEQ  END2        No, go back to write            *
END4B  LI   12,>Ø3ØØ    Yes,set VDP counter to 3        *
       JMP  END2        And now go back to write        *
END5   CB   1,@E        Is it "E"?                      *
       JNE  END6        No, go go                       *
       ABS  6           Is line counter already Ø?      *
       JEQ  END5A       Yes, then set at tcp last page  *
       AI   6,-21       No, subtract 21 lines           *
       JGT  END3D       Still past top 1st page         *
       JMP  END3C       Ø or belcw, go to reset         :
END5A  MOV  4,6         Set tcp of last page            *
       JMP  END3D       And go to auto repeat check     *
END6   CB   1,@FCTNE    Is it "FCTN E"?                 *
       JEQ  END7        Yes, go cn                      *
       CLR  12          No other keys,clear flag        *
       JMP  END3        And go back to KSCAN            *
END7   ABS  6           Is R6 already Ø?                *
       JEQ  END3        Yes, just go back to KSCAN      *
       DEC  6           No, decrease line counter       *
       JMP  END4A       And go to auto repeat check     *
```

```
ENDEND LI   Ø,>Ø2ØØ     Reset original screen image tabl
       BLWP @VWTR
       LI   Ø,>Ø1EØ     Reset 32 col.mode for Basic
       BLWP @VWTR
       MOVB @WS+1,@>83D4 Remember it needs to be saved
* Next 2 instructions return to original colors
       LI   Ø,>Ø7F4             if needed (your choice)
       BLWP @VWTR       F4 are colors in 6K XBASIC
       LI   Ø,START     Reload the ISR hook
       MOV  Ø,@>83C4
       CLR  Ø
       MOVB Ø,@STATUS   Clear GPL status byte
       LWPI >83CØ
       MOV  @SAV13,13   Restore the lost registers!
       MOV  @SAV14,14
       MOV  @SAV15,15
       RTWP             And back to basic!
************************
* SUBROUTINES
************************
* Clears PROBUF to spaces (without basic BIAS)
CLRBUF LI   3,>2Ø2Ø
       LI   4,PROBUF
       LI   5,2Ø        2Ø words=4Ø bytes
CB     MOV  3,*4+
       DEC  5
       JNE  CB
       RT
* adds the basic BIAS to all 4Ø positions of PROBUF
ADD6Ø  LI   1,PROBUF
       LI   2,4Ø
A6     AB   @BIAS,*1+
       DEC  2
       JNE  A6
       RT
* MOV34 is wrapped into MOVNUM routine below            *
* Read a record,assume a string with length byte is
* first,get LEN into R2 and INC RØ
RECRD  LI   Ø,PABLOC
       LI   1,>Ø2ØØ     READ OP code
       BLWP @VSBW
       MOV  @PABPT,@>8356
       BLWP @DSRLNK
       DATA 8
       JEQ  RR1         Error,might occur if 127 files *
*                       or 114 subdirectories.so to end*
       LI   Ø,VBF1      location of read buffer
       BLWP @VSBR       read 1st byte (length)
       INC  Ø           Beginning of string
       SRL  1,8         To LSB
       MOV  1,2         Put it in R2
       RT
RR1    B    @END1       Get past reading routine       *
********************************************************
* NOTE!!!!! The entire routine from here to the comment *
* line beginning "MOVE 21 LINES" is deleted and replaced*
* by the following routines.                           *
********************************************************
* Takes a Fl.Pt. # at VDP address specified in RØ, and *
* places it as an integer (word value) in R3           *
GETNUM LI   1,FAC       First we need the Fl.Pt. #     *
       LI   2,8         in FAC,for the next routine    *
       BLWP @VMBR       So,put it there                *
       LWPI GPLWS       Need GPLWS                     *
       MOV  @CFI,Ø      The Convert Floating Pt.# to   *
       BL   *0            Integer routine we found before*
*                         It takes a Fl.Pt.# at FAC and  *
*                         converts it to an integer word *
*                         value, placed at FAC           *
       LWPI WS            Now back to our WS             *
       MOV  @FAC,3        And save the integer in R3     *
       RT                                                *
* Takes an integer word value in R1 and first converts   *
* to ASCII decimal representation at NUMBUF, right jus-  *
* tified to 5 places(MOVNUM to GD3).Then it moves that   *
* sequence to CPU at location specified by R4. The small*
* section MOV34 at end moves R5 bytes from "at" R3 to    *
* "at" R4.                                               *
MOVNUM LI   3,10000       Max value has 5 decimal places *
       LI   5,NUMBUF      Where the number will go       *
GD1    C    1,3           Is R1>=current power of 10?    *
       JHE  GD2           Yes,go to processing           *
       MOVB @SPACE,*5+    Place a " " and advance counter*
       CLR  2             Get ready for division         *
       DIV  @D10,2        Places quotient(next lower pow-*
*                         of 10 in R2,remainder(=0)in R3 *
       MOV  2,3           Ready for next loop            *
       CI   5,NUMBUF+4    If R1=0,can't go beyond NUMBUF *
       JEQ  GD2           Jump to process (5th dec.place)*
       JMP  GD1           Try again                      *
GD2    CLR  0             Ready for division             *
       DIV  3,0           R3=current power of 10. Thus   *
*                         places 1st decimal digit in R1 *
*                         and remainder in R2            *
       AI   0,>30         Make # in R1 an  ASCII #       *
       SWPB 0             Place in MSB                   *
       MOVB 0,*5+         Place in NUMBUF,advance counter*
       CLR  2             Ready for division             *
       DIV  @D10,2        Next lower power of 10 in R3   *
       MOV  2,3           Save it for next loop          *
       CI   5,NUMBUF+4    Have we gone far enough?       *
GD3    JLE  GD2           No, go for more                *
       LI   3,NUMBUF      Source for next routine        *
       LI   5,5           5 bytes to move                *
* This part moves R5 bytes from R3 to R4                 %
MOV34  MOVB *3+,*4+
       DEC  5
       JNE  MOV34
       RT
* MOVE 21 LINES FROM BIG BUFFER POINTED
* TO BY R9 TO VBF3
* The next 7 lines replace the part originally at END2   *
SCRO   MOV  6,8           The page number                %
       MPY  @D40,8                                       *
* In the instruction MPY @ADDRESS,RY the result will be %
* in the 2 word sequence RY,RY+1 as a 32 bit number.What
* was in RY+1 before is wiped out.Thus in this case,R9
* will contain the result of the multiplication.
       AI   9,>1400       Now know which line in buffer  %
       LI   3,VBF3+120    Start on 4th line of screen    %
       LI   1,PROBUF      To transfer one line at a time
       LI   2,40          40 bytes
       LI   5,21          21 lines
SR1    MOV  9,0           Exact location in buffer
       BLWP @VMBR         Read to PROBUF
       MOV  3,0           Screen location
       BLWP @VMBW         Write on screen
       A    2,9           Change buffer location
       A    2,3           And screen location
```

```
       DEC  5             Go back for more
       JNE  SR1           If any!
* From here to RT checks for the auto repeat loop. Note *
* that the delay takes place BEFORE the next call to    *
* KSCAN, so that you can let go in time.                *
       ABS  12            In loop?                      *
       JEQ  SR2           No, just return               *
       MOVB @D10,@>8379   Yes, clear the VDP counter    *
DELAY  LIMI 2             Enable interrupts             *
       LIMI 0
       MOVB @>8379,10     Check the counter             *
       C    10,12         R12 was previously set        *
       JLT  DELAY         Not there yet, go back        *
SR2    RT                                               %
*********************
* DATA AND BUFFERS
*********************
DSKPAB DATA >000D,VBF1,0,0    INTERNAL,RELATIVE,FIXED   %
       BYTE 0             Note change in DSKPAB         *
DEVBUF BSS  40            And location of DEVBUF        *
DEV    TEXT 'DEV-NAME:'
* COUNT the characters carefully in TIT1-5!             *
*             0         1         2         3         4
TIT1   TEXT ' DISKNAM=            AVL=      USD=      ' %
TIT2   TEXT '    FILENAME  SIZE    TYPE         P     ' %
TIT3   TEXT '   ---------- ---- ------------- -       ' %
TIT4   TEXT '                SUBDIRECTORIES        -  ' *
TIT5   TEXT '                     FILES               ' *
TYPES  TEXT 'DIS/FIX DIS/VAR INT/FIX INT/VAR PROGRAM'
       TEXT ' SUBDIREC'                                 *
DSK1   BYTE 5                                           *
       TEXT 'DSK1.'                                     *
       EVEN                                             *
WS     BSS  32
NUMBUF BSS  6                                           *
PROBUF BSS  40
TOTAL  BSS  2
SAV13  BSS  2             SAV4 and SAV11 never used     %
SAV14  BSS  2
SAV15  BSS  2
COUNT  BSS  2                                           *
HARD   BSS  2                                           *
AVLSEC BSS  2                                           *
TOTSEC BSS  2                                           *
CFI    BSS  2                                           *
BIASCK DATA 0
D10    DATA 10            D100 was deleted, and these   *
D40    DATA 40            two were added                *
PABPT  DATA PABLOC+9                                    %
CTRLC  BYTE 131
READ   BYTE 2
CLOSE  BYTE 1
H40    BYTE >40           H83 never used                %
ENTER  BYTE 13
E      BYTE 'E'
PERIOD BYTE '.'+>60
X      BYTE 'X'
BIAS   BYTE >60
FCTNX  BYTE 10                                          *
FCTNE  BYTE 11                                          *
SPACE  BYTE ' '                                         *
FF     BYTE >FF                                         *
* remember to add DSRLNK, GPLLNK, and the VDP utilties  *
* here, and add the END directive at the end!           *
```

LI TOPICS

by Howie Rosenberg

REMINISCING

As one of the early owners of a TI 99/4(without the A), I hope you forgive me if I flash back to those early days from time to time.

At the time I bought my first TI computer, little was available in the way of software. My first module was Chess($79) followed by the only available game(Video Games). As no third party software at all existed, the only way to really make use of the machine was to write your own code. This was a blessing in that we non programmer types learned how to program out of necessity. In my case, I wrote a number of games primarily to show my two sons that I had truly purchased a useful machine. One of my sons asked(challenged) me to write a game similar to Breakout. He showed me the game at a local arcade, running at blinding speed. As Extended Basic had not yet been released, all we pioneers had was console Basic, there was no way to even remotely approximate Breakout. I told my son he would have to wait for the announced XB with its sprites. When XB finally was released, it did not take long to discover the real time sprite cooincidence problem. When using CALL COINC, a hit results only when coincidence occurs while CALL COINC is executing. Tight loops and large tolerances were the order of the day. Games had to be "invented" which could work under the existing environment. Breakout, of course, was out of the question. TI had already announced another gadget, the MINIMEM module which would allow programming at the assembly level so attempts at Breakout type arcade games was pushed aside until the release of MiniMem. In the meantime several third party companies started producing software. One which immediately gained great popularity was Millers Graphics whose early games were far superior to anything available in Extended Basic. What Miller did, and which still impresses me today, was relatively simple (most elegant, creative ideas are simple). In dealing with the sprite coincidence problem, Craig Miller rather than try to attempt to deal with the problem by conventional means, tight loops and wide tolerances, instead, predicted whether collisions would occur. This was a simple problem in physics, in general, and was done quite rapidly using simple equations. The trajectory was then simulated after the path and collisions were determined. Such a simple idea yet so elegant. This technique and many others which were created by Craig Miller appeared in Millers Sprite Guide which, to this day, is a classic and a must for all programmers in Extended Basic on the TI 99/4(and 4A).

Following the Sprite Guide, Craig started the Smart Programmer which was a source of unique ideas and the results of his probing into the hidden corners of the machine. Late arriving users know Miller Graphics(now MG) as a source of several fine products and excellent programs but I am sure many cannot understand the respect and gratitude felt by we the early users who lived through those days in the software/informtion desert.

FORTH BITS 6.... On Changing FORTH Definitions

You may want to look over chapter nine in Starting Forth. I will briefly summarize a few of the salient features of the language as they apply to this subject.

FORTH words are made up of several fields. The name field contains the name of the word. 31 characters can be used in TI FORTH. Next in order is the Link field which contains the address of the name field(NFA) of the previous word in the dictionary. The dictionary thus is a linked list. Next is the code field which contains an address to start execution in run time for different word types. All higher level FORTH words,in TI FORTH, contain the address -7CCC(Hex), All Variables the address -4616, and constants -450E. Words in machine code have unique addresses which indicate the location of the start of execution. Last is the Parameter field which, in the case of words of which we are concerned(higher level FORTH definitions), contain a sequence of addresses of the words in the definition which are to be executed. The last word in a Parameter field is always -5BBA which is the exit routine (compiled by ;). The FORTH interpreter is an endless loop which takes the next word in the input stream, searches the linked list comprising the dictionary from the last entry backwards, and causes execution of the word required. If the word is not found in the dictionary, and it is not a number, an error message is returned.

Why would one wish to change a FORTH definition? Some definitions one cannot change with impunity. Suppose one tried to change a definition such as +. This is used in so many other words in the language that a little reflection should convince you that the language is bound to crash should one attempt such a change. Modifying words should be restricted to definitions in higher level words which you have defined in applications and which are modified temporarily while in the process of debugging, as an alternate to changing the code and recompiling. I have found this ability to be quite a time saver.

If one compiles the word : + - ; subsequent entries such as 1 1 + . will result in 0 rather than 1. Great to confuse your friends but you have not changed the FORTH word +. The interpreter searches from the top of the dictionary and encounters the new definition for + and executes its code which is -. The interpreter thus never reaches the original word + but all words using + prior to this new definition have alreay compiled the address of the code for the original +. If this were not so there would be havoc as all definitions using + would be incorrect. Can you figure out how to redefine - to be equal to the old + after + has already been redefined? Obviously :- + ; won't work as + has been redefined prior to the new definition and - will still equal - . One answer is at the end of this column.

Twe methods of changing definitions come to mind. In the first method a new word is defined and then substituted for the old word by modifying the links in the link list. Thus if there were a sequence of words W1 W2 W3, and the word to be changed was WØ, the link field of word 3 would be changed to point to the name field of WØ and the link field of W2 made to point to the name field of W1. This method while apparently straightforward has a few pitfalls. Suppose that one first changes the W3 link field to point to the new defintion which is also named W2. This is readily accomplished with the aid of ' (tick) which returns PFA(Parameter field address) to the stack and with LFA(Link field address), and NFA(name field address) which with PFA on the stack return LFA and NFA respectively. Thus ' W1 NFA ' W2 LFA ! links the new W2 to the old W1. The only problem is that the interpreter now bypasses all previously defined words between W1 and the new W2 including W3. If on the other hand, one executes ' W2 NFA ' W3 LFA !, then a loop is created where interpreter executes words from new W2 down to W3 which then points back to new WØ. No words below W3 can be executed and control of FORTH is lost. One way around the problem is to determine and record the value of W3 LFA prior to linking W2 to W1. After linking W2 to W1 execute ' W2 NFA ' (recorded value) ! . This method is a bit cumbersome but can be used to modify any word even machine code words in the kernel(not recommended).

Another method usually easier to apply, is to move the words at the new word PFA to the old word PFA. At first glance it would seem that one would need determine how many words to move by checking for the occurence of -5BBA (the code word for the exit routine) in the words following PFA. Thus a sequence such as ' NEWWORD @ . ' NEWWORD 2+ @ . etc until -5BBA followed by ' NEWWORD ' OLDWORD (number of Bytes to move) CMOVE would accomplish the change. All previous code would now point to the new code. The word NEWWORD must be left in the vocabulary. This idea can be greatly simplified by simply defining a new temporary word : TEMP NEWWORD ; The PFA for this new word contains only two words. First a word pointing to NEWWORD followed by the pointer to the exit code. After moving this code to OLDWORD, it may be forgotten. It has the added advantage of allowing the code for NEWWORD to be longer than the code for which it is substituted.

Thus to change a word simply do the following. Write a new definition : NEWWORD (type in your new code here) ; Now create a temporary word : TEMP NEWWORD ; Execute ' TEMP ' OLDWORD 4 CMOVE FORGET TEMP and the job is done. This procedure is often a lot faster, during debugging, than recompiling many screens. Of course when the debugging job is completed you would then go back and change code by more traditional means.

By now you probably see that one answer to the question about - is simply to define a word : TEMP + ; PRIOR to the following : + - ; : - TEMP ;

FEST-WEST
=========

by Peter Hoddie, (C)opyright 1987

A Report on the LA Show
(and some Random Thoughts)
May 25, 1987 2:Ø3 AM May be reprinted with full credit

This file is intended as a rcport on Fest West, the TI computer show held in Los Angeles on May 16 and 17. The show was held at "The Shrine" in LA as part of a larger computer show. The main floor was occupied by all sorts of computer dealers selling mostly IBM stuff and accessories. The balcony was shared with Amiga taking one side and the TI 99 taking the other. Because the show was part of a larger event TI owners had an opportunity to see how the other half lives, while all those with PC's could see what they missed out on. Attendence at the TI portion of the show was difficult to gauge because of the large numbers of people wandering in who didn't own TI's. A reasonable guess would be between 4ØØ and 5ØØ people. The show was run by the LA 99'ers, and coordinated by Terrie Masters, former president of the group, now vice-president. Rumor has it that the show may be in Las Vegas ne..t year.

MG (formerly Millers Graphics) was there in full force selling their many fine 99/4A programs as well as showing their interesting Turbo XT which is being marketed by Triton. The XT uses the TI keyboard, and because of the limited number of keys on the 4A, the largest keyboard strip (I think it has _4_ levels) I have ever seen comes with the XT. MG was also showing Super Extended BASIC which is an upgrade to TI's Extended BASIC which is being handled by Triton. Super Extended BASIC is based on the Danny Michael modifications that appear in GRAM Kracker Utilities I along with a number of new CALLs added by Mike Dodd. I believe the cartridge is selling for $6Ø, and if you don't have a GRAM Kracker (so you can use GK Utilities I) I would strongly recommend that you consider purchasing a copy - the new editing and line moving capabilities are worth the price alone. MG was also promoting their VID program which is a video tape database program for PC's. It is quite a good program, as one would expect from MG, but since it is for _those_ computers I won't go into any more detail.

Bytemaster, represented by Richard Mitchell, was in attendance. Bytemaster used to publish Super 99 Monthly and now publishes the Smart Programmer, the premier publication for anyone who programs on the 4A. It features articles by Richard, Mike Dodd, Craig Miller, Doug Warren, Mariusz Stanczak, and many others. Topics such as GRAM Kracker and Forth which don't receive much attention elsewhere are given regular treatment here. Smart Programmer has been behind schedule as of late, but Richard is working hard to get back on track, and the quality of every issue has been outstanding. Bytemaster has recently gotten into software as well. In addition to the 2 disk set, "Best of Super 99 Monthly," they are preparing to release a package called String Master which is a must for the Extended BASIC programmer who works regularly with string arrays. Bytemaster is now distributing Doug Warren's extraordinary Explorer program, which was formerly carried by MG. The program now comes on 4 flippies, and is unprotected. There are versions for GRAM Kracker, Extended BASIC, and Editor/Assembler. The complete manual is on disk. I might add that the program is mostly Geneve compatible as well. It is still $25, and well worth it, to have a version of the program to put on your utility disk.

DataBioTics was there, represented by Bill Moseid. They were showing many of their programs, many of which I had never seen before. They have a version of Forth called Super Forth which was worked on by Edgar Dohmann (of Super Bug fame). It is a very fast, well documented version of Forth, that among other things supports a Winchester hard disk. It requires the use of a "super cartridge" such as DataBioTics' Super Space to work, and as such is available provides the Forth programmer with more room to work. They were also showing the shell of a program called SAM (that stands for something but I can't remember the details now). The program is a complete disk manager, including support for a hard drive. The program will be written in 100% assembly and can be used with both the old MYARC personality card (with new EPROMs) and the new MYARC hard drive controller. This program should prove to be quite popular because the utilities supplied with the MYARC hard disk are written in a combination of Extended BASIC and assembly and are extremely slow and painful to use. They were also showing Todd Kaplan's new WordWrite program which is a complete rewrite of the TI-Writer editor with many nice features including more memory. The program fits in a cartridge, which can include a built in printer port, and can use a cassette to load and save files. If you know anyone with a bare bones TI system looking for a word processing program, this is the only serious choice.

Dijit Systems was at the show, demonstrating their analog and TTL RGB display conversion kits for the 4A. These go for about $100 and let you use a high quality display with your 4A. I suspect the analog option may become quite popular as more people buy analog monitors to use with their Geneve. Dijit was also showing a prototype of a video card they are working on. This card allows for 80 columns, and is based around the 9938 chip that is used in the Geneve. It should be available in the next couple months for around $200. From what I've seen, it looks to be a better supported product than the 80 column card from Mechatronics.

A little known company by the name of Pesaca came down from San Francisco to show their programs. CharMat and Print Designer are their two current offerings and both are well documented, full featured, graphics type programs. Their main pitfall is that they only work with certain printers (although Tom Freeman and I didn't have much trouble in converting some of the programs to run on an Epson compatible) and they are a bit pricey. If you have one of the printers that is supported, I strongly recommend these programs. Details can be found in the last few issues of MICROpendium.

MYARC was present as represented by their west coast distributor Les Merryman. Since Lou Phillips was up at the Ottawa show, I (jph) was sent out there by MYARC to represent the company. Mostly we showed the Geneve 9640 and answered questions. The computer was very well receeved and all ten units that were available for sale at the show were sold. During the two days of the show many people brought over disks of programs that they wished to try out, and in over 95% of the cases the programs worked flawlessly. This made for the best demo because the people were 100% sure that nothing was rigged. They were seeing that this machine was compatible with their beloved 4A. I would dwell more on the subject of the 9640, except I think that we all know the details by now.

Since I was going to the show, I also dragged along products from Genial Computerware, Tigercub, Boston Computer Society, and the MAGNETIC user group. Tigercub had Nuts and Bolts 3 available as well as Tips from the Tigercub 4 and these sold well. The Boston Computer Society was selling disks from its popular software library and copies of Joyce Corker's excellent TI Writer Tips and Tricks booklet. Magnetic was selling laminated keyboard strips. Genial Computerware was selling subscriptions to volumes one and two of Barry Traver's incredible Genial Traveler diskazine, as well as XBasher by Mike Dodd, and XB:Bug, GRAM Packer, and the Horizon RAM Disk EPROM by me. Watch for reviews of these product in MICROpendium over the next couple months. I would say more, but then this would turn into an advertisement.

T.A.P.E. was there selling the Mechatronics product line including a very affordable EPROM programmer at about $140, the excellent Intern book by Heiner Martin, an 80 column card, a memory expansion for the their GRAM Karte (sort of a GRAM Kracker in a box), and the very popular "I <heart> My TI" baseball caps that Disk Only Software made famous at the Chicago show this past summer.

Rave 99 came in all the way from Connecticut and sold quite a number of their keyboards which allow you to use an IBM style keyboard with your 4A. They also now offer the product in kit form and without the keyboard for those who may already have an IBM type keyboard (such as 9640 purchasers). They were also showing their card which allows the speech synthesizer to be placed inside the expansion box for convenience on the 4A and required if you want speech with the Geneve. They were also quietly discussing some rather exciting future plans. Keep an eye on Rave 99; they are turning out high quality hardware products for the 4A.

There was a fairware booth, where you could get copies of all the latest fairware programs and make a contribution to the author on the spot. They were also selling a listing of nearly 200 fairware offerings for only a dollar. The table was run by Steve Mehr who did a great job in producing the list, and Ken Gilliland who has created some of the most incredible music programs ever written for the 4A. If you haven't heard (and seen) any of Ken's work send him a few bucks and some disks to the address below for some incredible material. He has done disks from Star Trek, The Wizard of Oz, South Pacific, Patsy Cline, Richard Wagner and more.

The LA group was premiering their new "Kracker Facts" book edited by Mike Dodd which contains page after page of useful information for GRAM Kracker owners. The material is by Mike, Tom Freeman, Craig Miller, Walt Howe, and others. It is available for $5 (and I suspect $1 for postage). They also were selling Tom Freeman's booklet and utility disk compilation of his incredible articles from the LA newsletter. The set is $8 and well work it.

There were a good number of local user groups with displays at the show, selling from their software libraries, but since I don't have a complete list I will refrain from listing them.

Because of space, I can't possibly write a paragraph about every person that was there, I am now going to name drop and just list the names of some of the many people who I met there. Tom Freeman, author of DISkASSEMBLER and president of the LA group; George Steffan, former vice-president of the LA group, and resident guru; Doug Warren, creator of Explorer; Bill Harms, author of Fas-Trans; Rodger Merritt, author of Print-It, the fairware alternative to Font Writer <gasp>; Ray Kazmer, author of several fairware programs; Joe Nuovolini, president, I believe, of the Colorado Springs user group; Maruisz Stanczak, author of the Forth column in the Smart Programmer; Fred Moore, software librarian for the LA 99'ers.

The show was a lot of fun. That is the true bottom line. Friday and Saturday nights the LA group sponsored a hospitality room at the hotel, and a wonderful time was had by all. The best part of the show was meeting people - old and new friends. That is what this community is in a large part about. It has to be. Without a huge company looming over us to protect and provide for us, we have to stick together. This show was a success because of the spirit of the people who organized it, who came to display their products at it, and who came to see what is still available for their 4A computer.

A special thanks to Terrie Masters, Tom Freeman, and George Steffan for driving me around, giving me a place to stay, and in general making my stay in LA a wonderful time.

+++

Addresses (in no particular order)

Ray Kazmer, 13225 Azores Ave, Sylmar, CA 91342
Ken Gilliland, 543 Riverdale #15, Glendale, CA 912Ø4
Bill Harms, 6527 Hayes Ct, Chino, CA 9171Ø
J. Peter Hoddie, 12 Paul Revere Rd, Lexington, MA Ø2173
Genial Computerware, PO Box 183, Grafton, MA Ø1519
[except for the Genial TRAVelER diskazine, for which the address is Genial Computerware, 835 Green Valley Drive, Philadelphia, PA 19128]
MG, 1475 W. Cypress Ave, San Dimas, CA 91773
Bytemaster, 171 Mustang St, Sulphur, LA 7Ø663-6724
Dijit, 4345 Hortensia St, San Diego, CA 921Ø3
Boston Computer Society, One Center Plaza, Boston, MA Ø21Ø8
Rave 99, 23 Florence Rd , Bloomfield, CT Ø6ØØ2
DataBioTics, PO Box 1194, Palos, Verdes Estates, CA 9Ø274
MYARC, PO Box 14Ø, Basking Ridge, NJ Ø792Ø-1Ø14
Tigercub, 156 Collingwood Ave, Columbus, OH 43213

+++

And while I have your attention I would just like to make a few completely unrelated comments. In answer to the most popular question of the show "when will the Geneve be shipping," I can only say "real soon now." Actually sometime in the next week or two. So they tell me. And in answer to the second most popular question of the show "when will Font Writer II be shipping," I can only say that I plan to start clean up work on it as soon as I finish this file, so about 2 weeks, assuming I get that gate array from Mitsubishi . . . . who got Genial Traveler volume 1 number 6, and the John Calvin Traver disk that came with it, the Sort Experiment program on there by me doesn't work. It was sort of a combined foul up by Barry and myself. The corrected files are available on CompuServe and GEnie, and will be sent out with the next issue of Traveler. If you want them, or are not a Traveler subsrcriber and are looking for a very fast assembly language sort, that can handle any file type, up to 1ØØØ records or 24K of data, ascending or descending sorts, on up to eight fields, send a disk and return mailer with postage to the address below, or $3 (to cover the above, that $3 is not the requested fairware donation) to me at the above address. Complete source file are included. Thanks for reading this file, your eyes must be pretty tired by now. I know my fingers are. -jph

[Postscript by BAT, who was at Ottawa instead: Actually, the SORT program sent with GT #6 _does_ work. The minor bug (discovered by Tom Freeman) was that it could only handle single-digit entries for starting character entries (e.g., you could tell it to sort beginning at character 6, but not character 16, since it would ignore the first digit). The fix in the source code is a simple one. In line 721 of DSK1.SORT/EXP/S, you'll find this:

```
MPY @D1Ø,R1 multiply current value times 1Ø
```

Before that line, insert the following short line:

```
MOV R2,R1 Thanks you, Tom Freeman!!!!!
```

In other words, your source code for SORT/EXP/S should now look like this:

```
Ø721 MOV R2,R1 Thank you Tom Freeman!!!!! Ø722 MPY
@D1Ø,R1 multiply current value times 1Ø
```

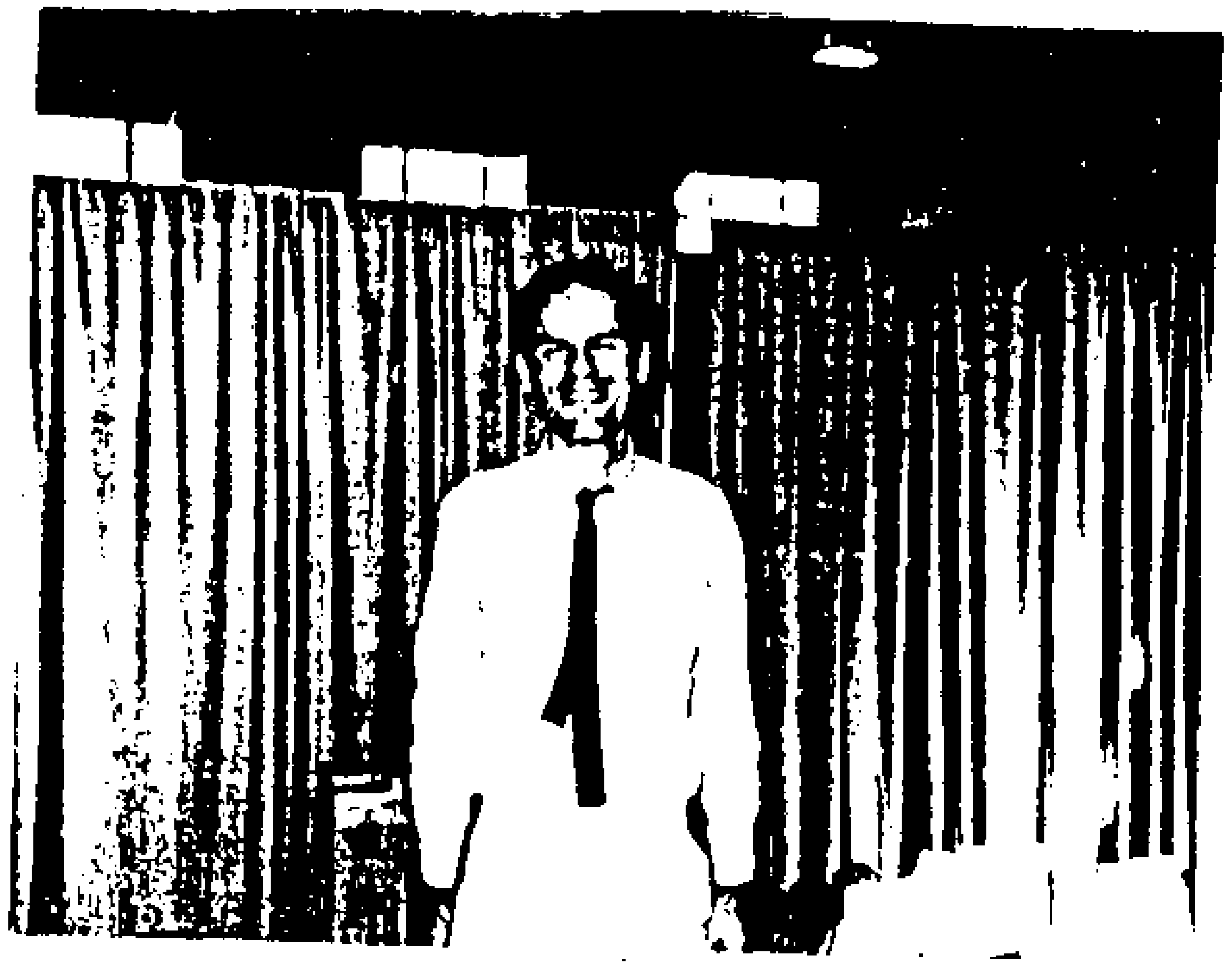
My mistake was omitting to include the program image file in SORT/ARC, but you'll have to reassemble it anyway. Here are the steps: (1) Assemble (the corrected) SORT/EXP/S to create SORT, (2) assemble SAVE/S to create SAVE, (3) Load SORT and SAVE into the Editor/Assembler option 3, linking to program name SAVE. That's all there is to it to create an all-a/l sort program to sort essentially any type of disk file (e.g., DV8Ø, IF2Ø, etc.) on up to eight fields, thus providing a sort program with much greater speed and flexibility than you may be accustomed to! Try it - you'll love it!

RAVE's John and Ken

Tricia Miller

"the three M's"

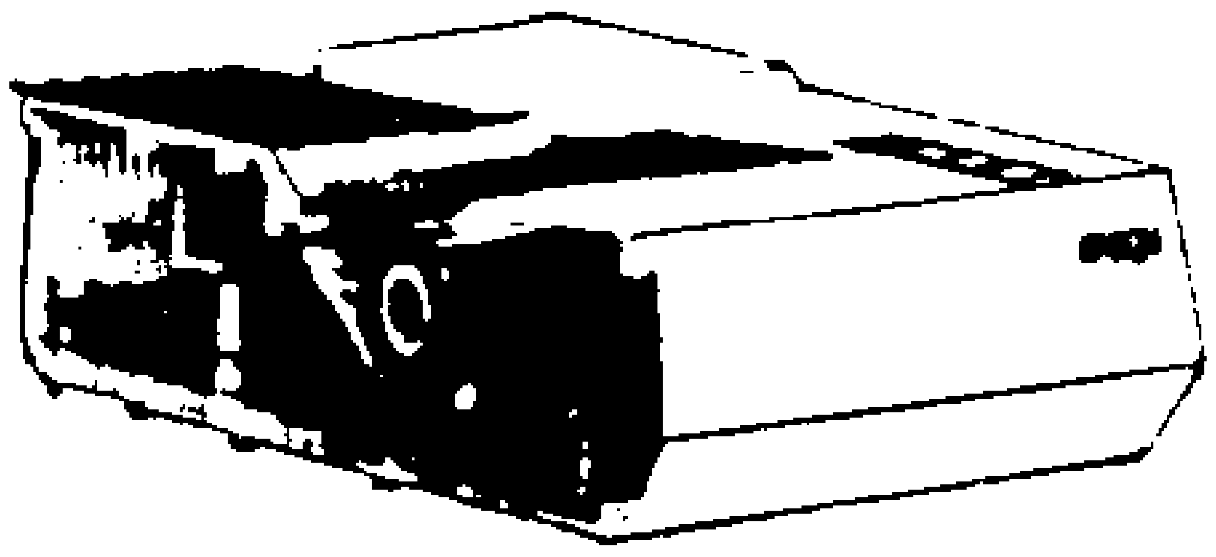
DIJIT's Tom Spillane



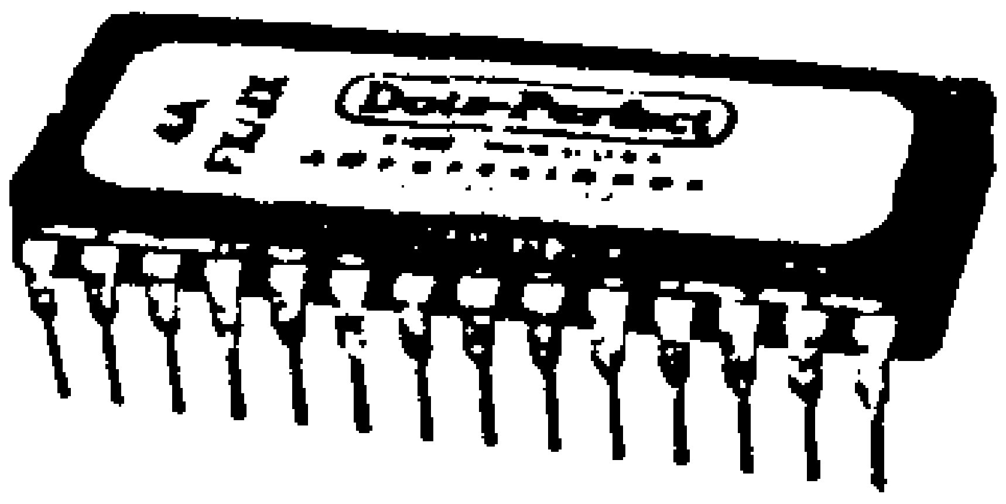
Data Systems' George Holod

Pesaca's Karl Leuhrman

You already own half of a great printer...



Are you still looking for the other half?



Look no further!

Introducing : Dots-Perfect™,
an upgrade kit for
Epson® FX, JX,
RX, and MX printers

sample of letters without Dots-Perfect
ABCDEFG
abcdefg

sample of letters with Dots-Perfect
ABCDEFG
abcdefg

g Sample of letter with Dots-Perfect

Sample of letter without Dots-Perfect g

Dots-Perfect™ is here at Super Discount Prices

Just $63.00 + $2.00 S/H

For orders send check or money order to:
Myra D. White
849 E. Bonita Ave.
La Verne, CA. 91750
(714) 592-6897 after 6PM PST

California residents
add $4.10 sales tax

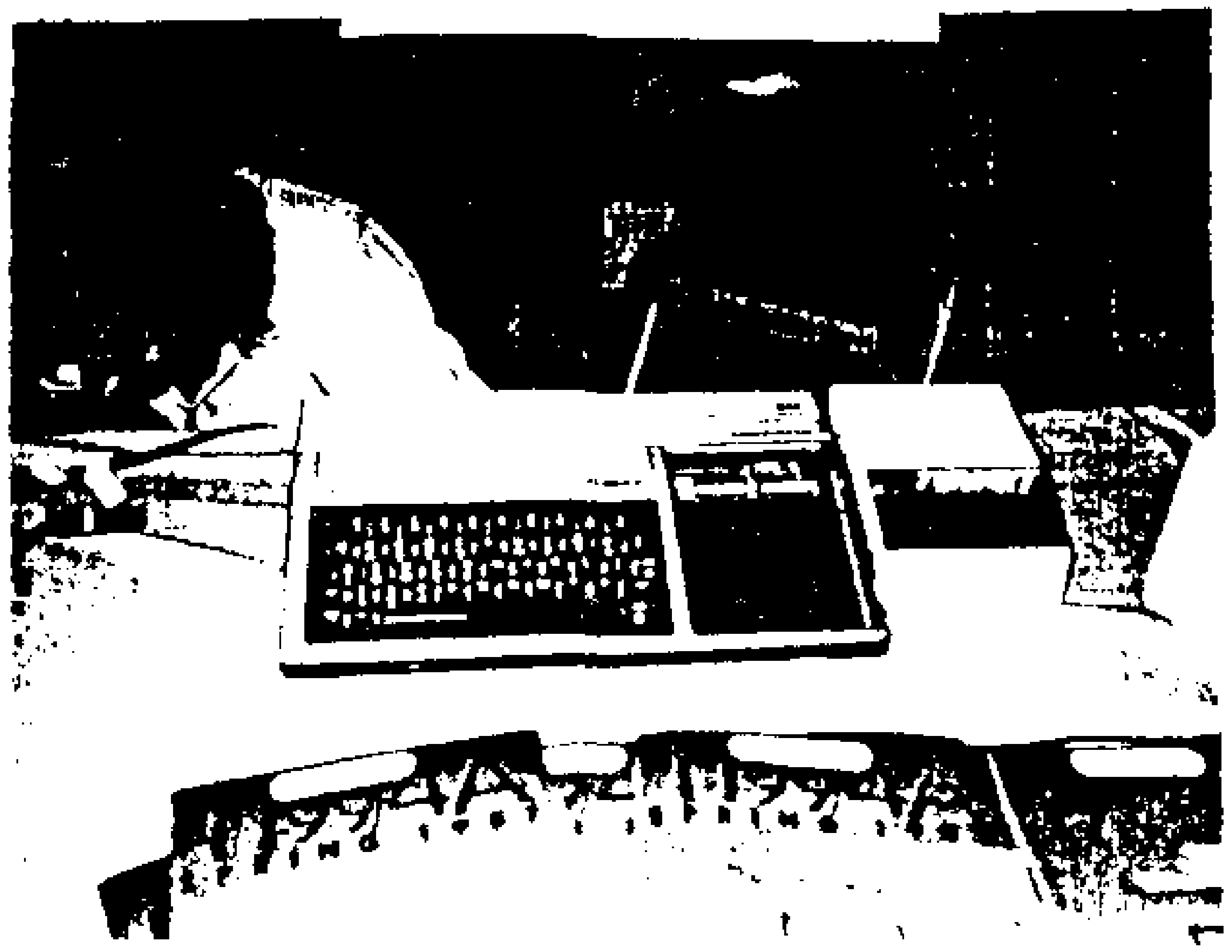
All orders shipped within 48 hrs of receipt

Steve Mehr and Ray Kazmer

Ken Hamai

Joe Nuovolini

Roger Merritt



Craig Miller

George Steffen

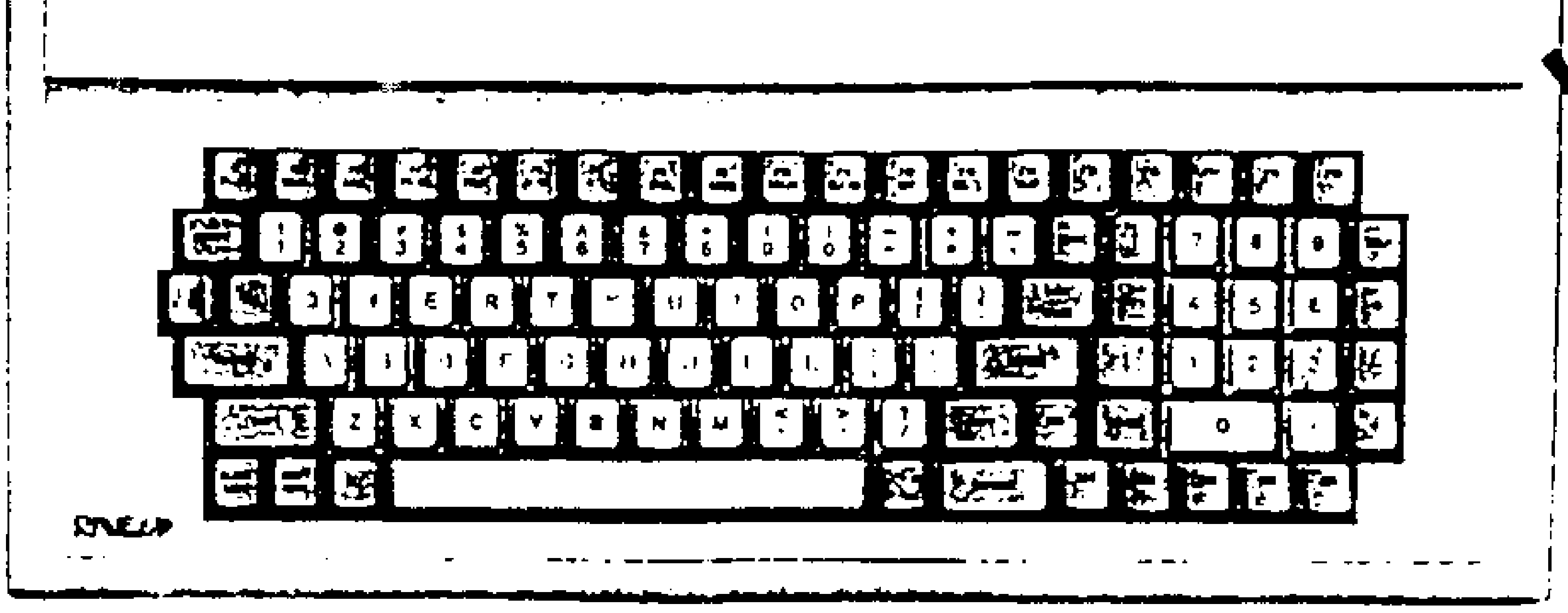
Ed May, and Gail Fair's back

Tom Freeman, Terrie Masters, & Steve Chalcraft

Tom Freeman
at GEnie International CO

MODEL 99/105 KEYBOARD ENHANCEMENT - $199.95

NEW!



INCLUDES custom keyboard.

THE FOLLOWING ARE STANDARD FEATURES ON THE MODEL 99/105 KEYBOARD ENHANCEMENT...

- * * INTERFACE LOGIC CARD WHICH INSTALLS EASILY IN CONSOLE WITHOUT SOLDERING.
- * * 105 KEY KEYBOARD WITH 24 FUNCTION KEYS.
- * * SUPPORTS SINGLE KEY ENTRY OF ALL TI-WRITER, MULTIPLAN, AND EDITOR/ASSEMBLER COMMANDS.
- * * ATTRACTIVE LOW PROFILE KEYBOARD WITH ADJUSTABLE TILT ANGLE AND 5 FT. CORD.
- * * DEDICATED NUMERIC KEYPAD WITH NUMERIC OPERATORS.
- * * DEDICATED CURSOR CONTROL KEYS.
- * * ENLARGED RETURN, SHIFT, AND CONTROL KEYS.
- * * OPTIONAL USER INSTALLED RESET AND LOAD INTERRUPT CAPABILITY FROM KEYBOARD.
- * * DETAILED INSTALLATION AND OPERATING INSTRUCTION MANUAL INCLUDED.

KEYBOARD INTERFACE CARD - $149.95

- * * Interface card installs in 4A console in place of old keyboard and allows use of any IBM PC/XT compatible keyboard with your 99/4A. Interface card supports all 83 standard PC/XT key codes.
- * * Comes with complete, easy to follow installation and operating instructions. NO SOLDERING REQUIRED.
- * * Optional user installed RESET and LOAD INTERRUPT capability from keyboard. Complete instructions included.

NEW!

KIT AVAILABLE

- * * Includes printed circuit board, complete instruction manual, and RAVE 99 custom chips. Hard to find hardware also included. Requires only assembly and about $10.00 in easy to find components to complete. $ 92.00

* IBM & PC/XT are trademarks of International Business Machines

some "RAVE" reviews!

..I wonder how I have managed all this time ..no longer have to go through those awful contortions...an absolute joy!! D.R. Harrisburg, PA.

.. the keyboard is everything you said and more...performs exactly as advertized...has operated flawlessly. J. C. Worthington, OH.

.. I would hate to go back to the original J. L. Hicksville, NY.

..having trouble restraining myself... nothing but superlatives.. I have stepped up to big time computing. J. W. Englewood, OH.

.. this keyboard is fantastic !! M.H. Colonge, West Germany

.. I played, I explored, I fell in love...by Gawd there is a Santa Clause. This thing is GREAT! .. absolutly no problem installing it. M.M. Tempe, AZ.

SATISFACTION GUARANTEED!

☆ RAVE 99 ☆

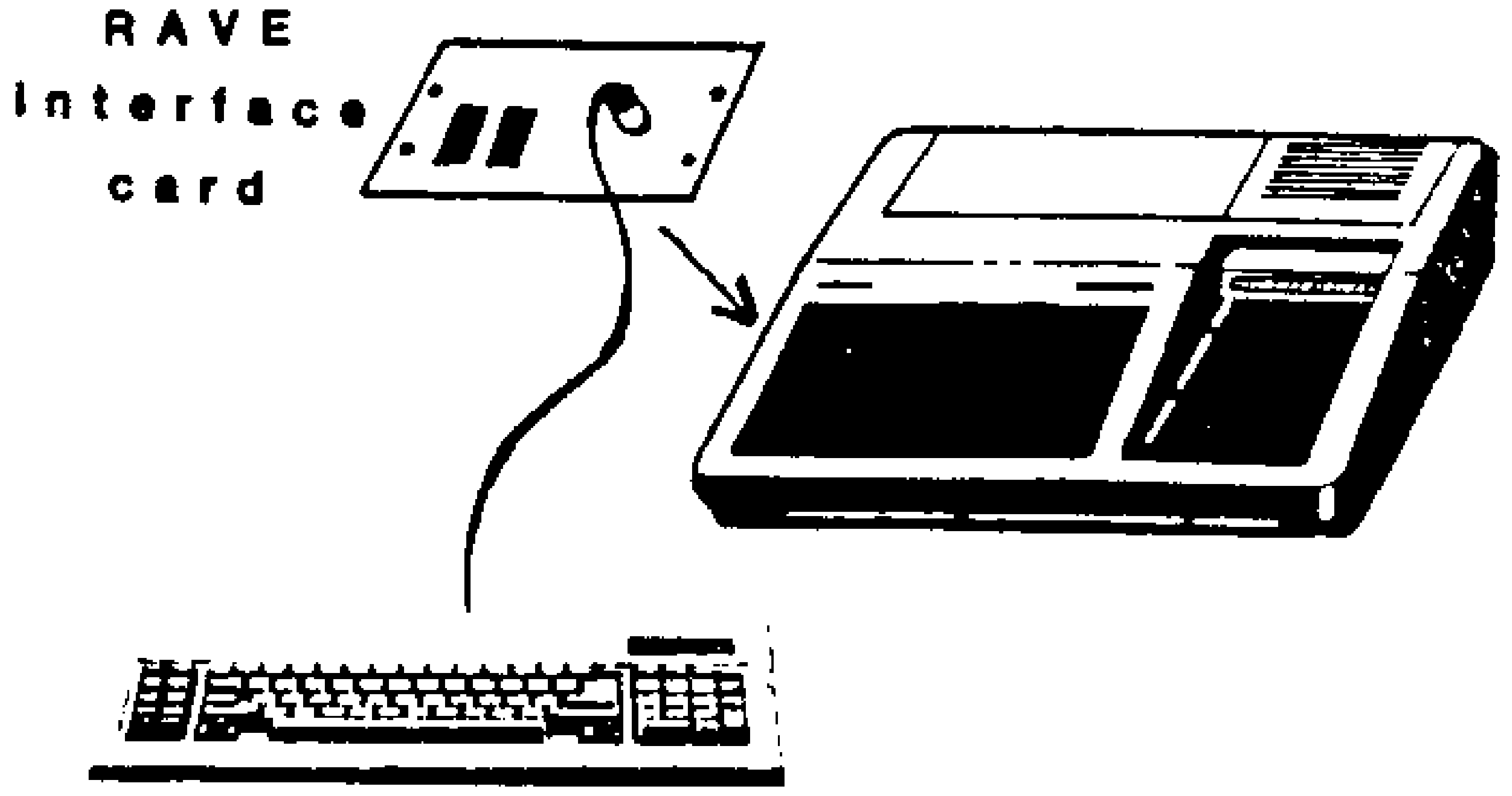
112 RAMBLING ROAD
VERNON, CT 06066

☆ (203) 875-9833 ☆

ADD 5% FOR HANDLING AND SHIPPING.
CT. RESIDENTS ADD 7.5% SALES TAX.

VISA & Master Card Add 3%

EASY INSTALLATION



MARKETPLACE

(the marketplace is a fund raiser for the club, that is, the "profit" goes to maintain the quality of this Newsletter. In general the price listed splits the difference between cost and retail. Please help your Club.)

MILLERS GRAPHICS	
DISkASSEMBLER	18.50
ORPHAN CHRONICLES (PRICELESS)	9.95
ADVANCED DIAGNOSTICS	18.50
NIGHT MISSION	18.50
GK UTILITY I	10.00
SMART PROGRAMMING FOR SPRITES	6.25
NEW RELEASES	
KRACKER FACTS	5.00
JOYPAINT	30.00
JOYPAINT PAL	7.50
UTILITIES DISK/DOCS (T FREEMAN)	8.00
PRE-SCAN IT! (J.PETER HODDIE)	10.00
GRAM PACKER "	10.00
FONT WRITER "	19.00
PRINTER'S APPRENTICE (M.McCANN)	19.00
MYARC	
RS232	Check
D/D DISK CONTROLLER	
128K RAM DISK/SPOOLER	for
512K RAM DISK/SPOOLER	
EXTENDED BASIC II LEVEL IV	discount
128K RAM DISK W/XBASIC II	
512K RAM DISK W/XBASIC II	prices
GENEVE 9640 COMPUTER	
INSCEBOT	
TI-ARTIST	15.00
DISPLAY MASTER	12.00
ARTIST EXTRAS	6.00
GENIAL COMPUTERWARE	
XBasher (MIKE DODD)	9.00
XB:Bug (J.PETER HODDIE)	9.00
MEGATRONICS	
EXTENDED BASIC II PLUS	72.50
INTERN (BOOK ON GPL)	16.50
128K GRAM CARD	227.50
HARDWARE & SUPPLIES	
TEAC 55BV DSDD DRIVES	90.00
DISKETTES DSDD	.50
64K EPSON INT. PRINT BUFFER	45.00
COLOR RIBBONS (EPSON)	4.00
BACK ISSUES	
SUPER 99 MONTHLY	1.25
MICROPENDIUM	1.25
SMART PROGRAMMER JUNE 1986	1.50
BEST OF NEWSLETTERS W/DISK	5.00
FORTH NOTES VOL 1-6 (2.50 EA)	10.00
BEGINNER'S FORTH NOTEBOOK	2.50
ASSEMBLY NOTES VOL 1	2.50
TECHNICAL AND BUSINESS BOOKS	5.00
SAMS BOOKS (VARIOUS)	5.00
SAMS BOOKS WITH CASSETTES	7.50

(please send your order to the CLUB address, not the Librarian, and add $1.00 per disk for postage and handling. CA residents add 6.5% tax).

NEW PROGRAMS FOR JUNE

4129 **ARCHIVER #2** Freeware by Barry Boone Compatible with 4104 ARCHIVER #1,and seems to be a little faster. ARCLOAD is the loader. (SSSD)30

4130 **LINKER** Freeware by R.A.Green. A tool for building assembler language memory image programs from tagged object. It makes this process simple and straight forward. (SSSD)233

4131 **DISK LABELS** Freeware by Dennis Porpora. Prints: 1 or 2 disk catalog side by side. Print disk catalog on Avery labels up to 70 files. Prints a labels for both side of a disk. prints an index card and an address label.)SSSD)284

4132 **CHAINLINK** Freeware by Walter Howe. This is an excellent solitare game using entire deck of 52 cards. A unique (to me) version of solitare that actually requires a little cogitation to beat it...you have to look at several moves in the future, like chess. Send him a ss/sd disk, mailer, postage, and a reasonable cash contribution if it appeals to you.

4133 **SUPERDIAL** Freeware by Roger Davis. Use with Fast-Term (on disk). 3 functions 1. PC PURSUIT DIALER. 2. LOCAL DIALING. 3 VOICE DIALING and BBS connection. (SSSD)310

4134 **PRO 99 BBS** Freeware by Mark Hoogendoorn. Improved by Roger Davis. This is a TI BBS with true TE2 transfer capabilities. Instruction on setting up and running 99 BBS system. (SSSD)342.

4135 **GRAPHIC LABELER** /// Steve McWatty RR#1 Kinburn,ONT KOA 2HO CANADA. To use Graphic use Dave Rose characters set. Prints 5 roll label with 1st line double emphasied, auto center (SSSD)63

4136 **DISK HACKER** /// Will McGovern /// 215 Grinsell St. Kotara, NSW 2289 AUSTRALIA -TI Controller only- A powerful disk utility to examine the actual format of each track of a disk. Single or double sided. Olso has cassete and A/E program loader. (SSSD)217

LIBRARIAN FRED MOORE 7730 EMERSON AVE. LOS ANGELES CA 90045

The LA-FEAST-WEST

by the Pomona User Group sec., Bill Harms

The second annual southwestern United States TI Home Computer FEAST really was a feast: of friendly folks helping others, of hardware, software, firmware (both new and used).

The LA User Group sure brought out a lot of TI enthusiasts for a well attended two days of questions and answers, demos, displays of all sorts of neat new, more powerful hardware and software.

Having a booth at the feast was well worth the fee and effort. It was nice to support the TI Home Computer community. We were able to hand out several fliers about our Pomona club and answer questions and sell used items. Plus, we got to talk with lots of famous persons, who have invented software, hardware, firmware and/or have promoted the TI-99/4A user interests.

The LA group sure put-in the power with Fred Moore furiously copying library software from their hugh library and Terrie Masters, George Steffen, Tom Freeman and Gail -?- all working hard, and esp. those two folks processing the registrations and raffle tickets. Boy were they all busy. I was even lucky enough to win the 2nd prize, an Oscar, which is a bar code reader device for quick entry of basic programs!

Speaking for the Pomona club, it was a smashing success, esp. since a truck broke the big entrance door Sunday PM. We even found a 2nd table for the club booth on Sunday AM on which to display all the stuff we had available.

It was a valuable get-together for active TI computerists and others who might be interested in using a TI.

2039 **UTILITY #1** $5.00 14 Utilities programs from England -BASIC SPIRTS, CHDEF(draw sprite), CHARPAT(define all keys), CHARMERGE(charpat in merge format), CHARSAMPLE(demo of charpat), CHECKLIST(find any variable), CURFLIP(alter cursor to show color), ART, HEXDECHEX(converts Hex to Decimal and vice-versa), MM>DSK(load programs into Mimi Memory from disk), DISPLAY-AT(useing TI Writer to create a blank screen), PILOT/INST(brief instrction for "Pilot"), QUICK-SORT(add to existing file for sorting), PROG/CHKR(compare 2 program listing), REMDIVIDER(create 2 new program from your old 1=all/REMS 2= no REMS), SCREEN-MAP(locates and compares screen addresses), SCREENDUMP(dumps anything on screen), STRFILE(relocate assembly programs in memory), KAMIKAZE(game), TINYTIPS(tips for basic programmer), TVTESTCARD(draws test pattern on screen). (SSSD)329

2032 **UTILITY #2** $5.00 Use Editor Assembly or Extented Basic. Two popular programs on one disk TI-WRITER and DM-1000. When useing E/A Load and Run OK TI no good CorpComp Controller. (SSSD)237

2041 **UTILITY #3** $5.00 X/B or E/A 12 Utitlties program OW/SLASH, CHANGE TO SECTOR 0, DISASSEMBLER, DISK FIXER, GOSUB/N, LABEL 99, LOAD SECTOR COPY, RAW INFO, RESTORE/N, SCREEN DUMP, CATALOG, VDP. (SSSD)338

2028 **UTILITY #4** $5.00 JAYS-PROGRAMS self load in XB, Speech, TE II, 32K, Printer,and Disk. - CATALOGER, CALCULATOR, ENVELOPE, PHONE #'S, SALES-SLIPS, CLOCK, WORD SEARCH, TIMER, SAY, BALANCE, Etc. (SSSD)215

2029 **UTILITY #5** $5.00 CHARACTERS SET By Jay Leber - 15 programs X/B - Change the characters set TI-Writer. Merge into your own program. (SSSD)146

2027 **UTILITY #8** $5.00 An excellent disk cataloguer by Stephen Shaw - To be called from your XB program with CALL LINK. A memory image file, Several character sets in various formats for use in your program. A X/B disk copier, and more. (SSSD)254

4043 **DISK HELPER** $2.00 Freeware by Jim Mekeel. A utility disk with disk manager,CATALOG PRINTER and DISK FIX/VAR READER. SSSD (265)

4066 **BILL'S MIXTURE** $2.00 Freeware by Bill Rodriguez. An interesting mix of original Catalogers and Sprite designers, along with one of the best Morse Code practice programs. Including the source codes for one or two of them, there are twelve different programs all written by a non professional programmer. Send a disk and a re-useable mailer, along with a couple of dollars for the whole disk of interesting programs. SSSD (325)

4117 **DISK UTILITIES** $2.00 Freeware by John Birdwell. X/B Provides the user with the means to study how data is stored on disk. Good for changing and editing the data to suit their purpose. Some of the files on disk Compare Disks, Print and edit Sectors, Find strings, Disk Report and Directory, Printer Setup, Prints and Edit Files, and many more items. (SSSD)211

4087 **ML/Utility** $4.00 Freeware by Art Green, Issue 2 Contains 6 assembly language utilities including 3 disk utilities, a terminal emulator and a printer set-up for dumping PrintArt files to your printer, with a disk to cassette added in this issue. DSSD (567)

June 1, 1987

L.A. 99'ers User's Group
P.O. Box 3547
Gardena, CA 9Ø247

Dear Editor:

I would appreciate your passing this inquiry to your club members. I'm interested in corresponding with anyone that has experimented with MIDI hookup to the 99/4A. MIDI stands for Musical Instrument Digital Interface which is a standard in the music industry much like RS232 in the computer world. In any event, if the term MIDI (pronounced mid-dee) doesn't get a response, consider it a dead horse.

Happy Computing,

Lane Douglas
1616 Richforest Dr.
Richardson, TX 75Ø81

PRODUCT ANNOUNCEMENT

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus, OH 43213
(614) 235-3545

Tigercub Software has released Nuts & Bolts Disk #3, containing another 14Ø subprograms in MERGE format. Contents include 19 screen character fonts, etc.; 17 screen display routines; 6 scren formatting, 8 plotting , 6 joystick and keyboard, 32 math, 4 time and date, 1Ø input and accept, 9 string handling, 15 file handling, and 9 miscellaneous routines. The 11 pages of documentation contain a programming example to demonstrate the use of eaach routine.

The three Nuts & Bolts Disks now provide a total of 34Ø subprograms which even a beginning programmer can merge into his own programs and use, almost like having another 34Ø CALLs available in Extended Basic. The price of all three of these disks has been reduced to $15 each, postpaid.

The four Tips from the Tigercub disks and the 18 Tigercub Collection disks have been reduced to $1Ø each, postpaid. The 13Ø individual Tigercub programs have been reduced to $2 each, plus $1.5Ø per order for cassette or disk and postage (minimum order $1Ø). Cassette orders will only be filled until stocks of blank cassettes have bee exhausted. Tigercub catalogs are available for $1, deductible from first order, until stocks are exhausted.

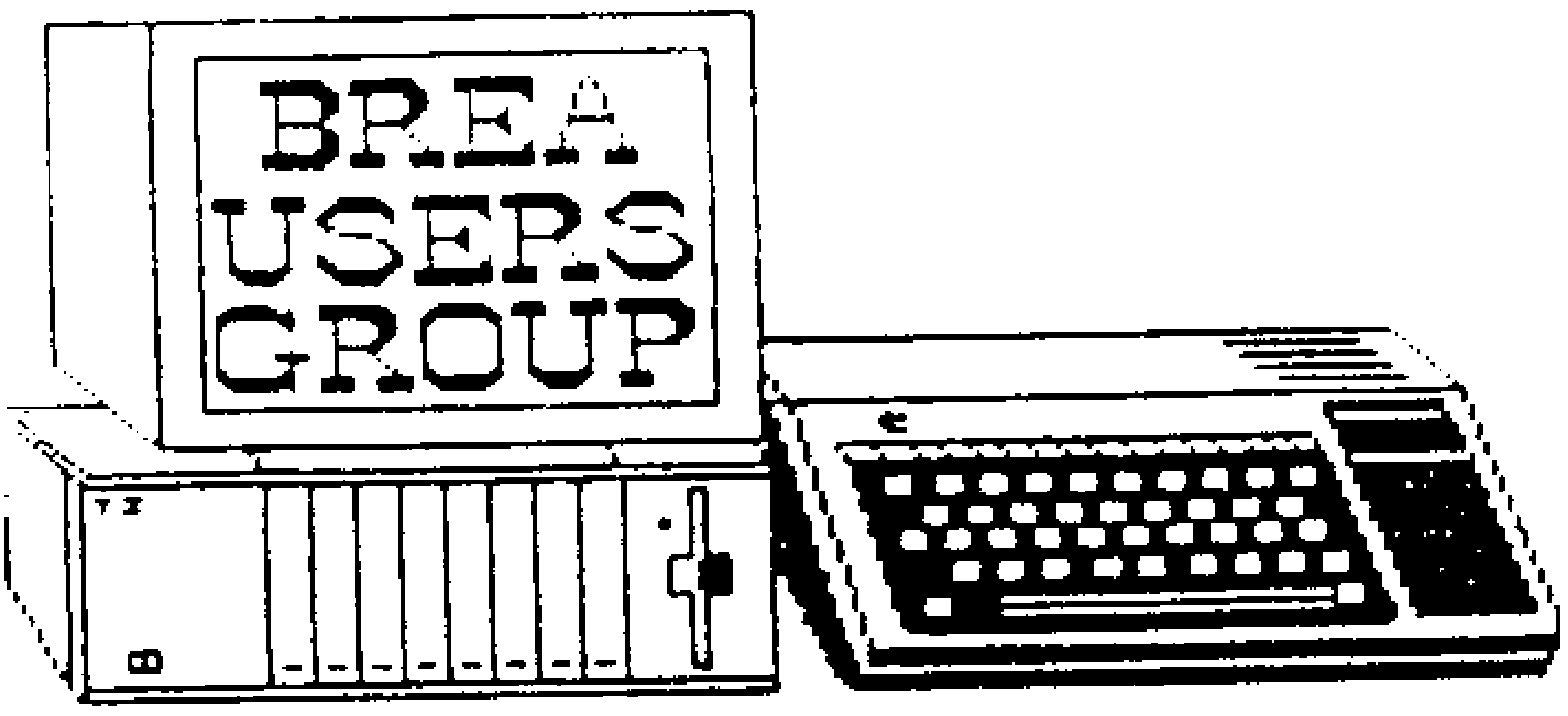
NUTS & BOLTS UPDATES

There is a major bug in FORMATTER on the first Nuts & Bolts disk. The last statement in line 2Ø171 should be SUBEXIT, not END. A minor bug in Nuts & Bolts #2 prevents using HIGHCHAR after HEAVYCHAR. To fix it, resequence HEAVYCHAR by RES 21ØØØ,1.

Some users have reported having problems with the subprograms which contain DATA, ON ERROR, or a flag routine. As is explained in the documentation of Disk #3, a READ statement will read the next unread item of DATA whether it is reading from the main program or from a subprogram, and whether the DATA is in the main program or in a subprogram, unless a different line item of DATA has been RESTOREd. Therefore, be sure to RESTORE the next DATA line to be read after you leave a subprogram which contains DATA - it is good programming practice, anyway, to RESTORE all DATA befre you read it.

When an error is encountered, program execution responds to the last open ON ERROR statement, whether that statement is in the main program or in a subprogram and whether the error occurs in the main program or in a subprogram. Therefore, be sure to restate any ON ERROR in the main program after leaving a subprogram which contains ON ERROR.

Many of the subprograms contain a flag routine immediately after the SUB, in the form IF F=1 THEN (line number) :: F=1. This speeds execution by skipping over initialization after the first CALL, but may make it impossible to reuse the subprogram. For instance, a redefined character set which has been cancelled by CALL CHARSET cannot be CALLed again. In many cases, this flag can be deleted. Or, the subprogram can be renumbered and renamed, merged and CALLed under a different name. Or, an additional parameter can be added to the SUB and the CALL to turn the flag on and off.



May 20, 1987

LA 99ers Computer Group
P. O. Box 3547
Gardena, CA 90247-7247

Attention: Tom Freeman, President

Dear Tom,

We want to thank you and your group for putting on the second annual (we hope) TI-Fest West. The members of our group who worked the booth felt that it was well organized. We only wish that there had been more seminars like they had at the Boston Fair.

In any case, we certainly felt that it was a success and you can count on us for as long as we can keep this TI-99 thing going.

Sincerely,

Kennett S. Hamai
V.P./Media Chairman
Brea 99er's Users Group
11508 Mollyknoll Ave.
Whittier, CA 90604

REMEMBER NEXT MEETING - Wednesday June 24, Torrance Public Library, 7 PM

CLUB OFFICERS

President:		Membership Chairman:	
Tom Freeman	(213) 454-1943	Ed May	(213) 644-6241
Vice President:		Librarian·	
Terrie Masters	(213) 271-6930	Fred Moore	(213) 670-4293
Secretary:		Library Assistants:	
Doug Moore	(213) 451-1069	Alan Whiteman	(213) 379-8031
Treasurer:		Chick De Marti	(213) 532-8499
Alan Whiteman	(213) 379-8031	Equipment Chairman:	
TopIcs Editors:		Joe Fierstein	(213) 377-9834
Terrie Masters	(213) 271-6930	Hospitality Chairman:	
Tom Freeman	(213) 454-1943	Myron Harms	(213) 675-3959

Membership in the LA 99ers, including subscription to TopIcs is $20.00 per year