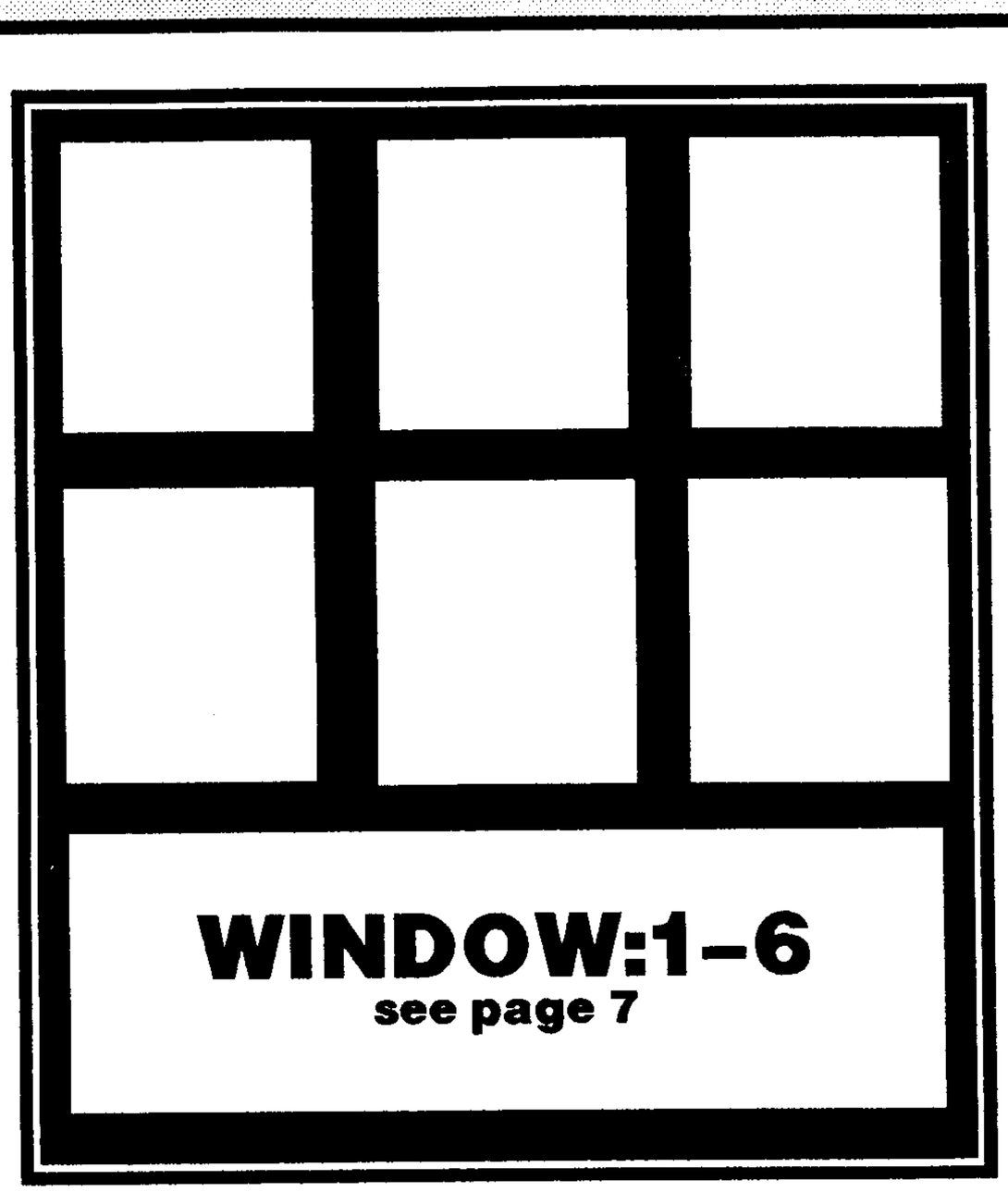
Covering the TI99/4A and the Myarc 9640

MICROpendium

Volume 12 Number 9 October 1995 $3.50



PC99 vs the Red Baron

Reviews of PGRAM Utilities V2.3,
Ian's Games, Schematic and Font Dumper

Report on the New England Fall Fair

CONTENTS

MICROpendium

MICROpendium (ISSN 10432299) is published monthly for $35 per year by Burns-Koloen Communications Inc., 502 Windsor Rd., Round Rock, TX 78664-7639. Second-class postage paid at Round Rock, Texas. POSTMASTER: Send address changes to MICROpendium, P.O. Box 1343, Round Rock, TX 78680-1343.

All correspondence should be mailed to MICROpendium at P.O. Box 1343, Round Rock, TX 78680. We cannot take responsibility for unsolicited manuscripts but will give consideration to anything sent to the above address. Manuscripts will be returned only if a self-addressed stamped envelope is included.

Foreign subscriptions are $40.25 (Mexico); $42.50 (Canada); $40.00, surface mail to other countries; $52 airmail to other countries.

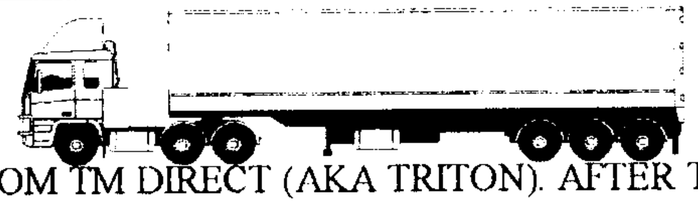
All editions of MICROpendium are mailed from the Round Rock (Texas) Post Office.

Mailing address: P.O. Box 1343, Round Rock, TX 78680.
Telephone & FAX: (512) 255-1512
Delphi TI NET: MICROpendium
GEnie: J.Koloen
Internet E-mail: jkoloen@io.com
John Koloen.....Publisher
Laura Burns.....Editor

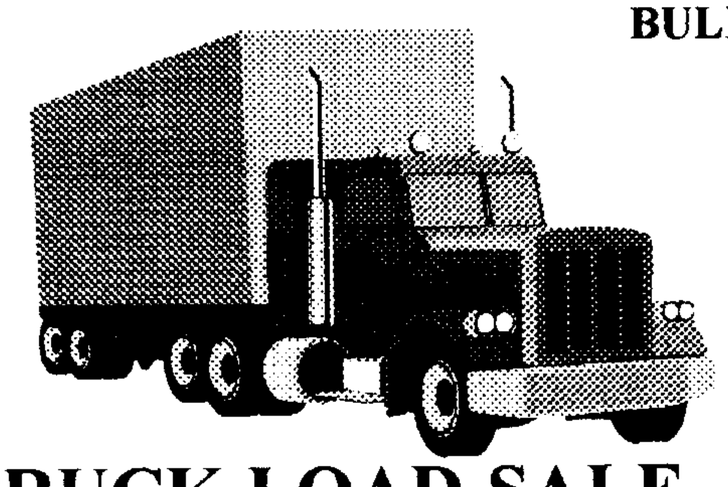
*READ THIS

Here are some tips to help you when entering programs from MICROpendium:
1. Most BASIC and Extended BASIC programs are run through Checksum, which places the numbers that follow exclamation points at the end of each program line. Do not enter these numbers or exclamation points. Checksum is available on disk from MICROpendium for $4.
2. Long Extended BASIC lines are entered by inputting until the screen stops accepting characters, pressing Enter, pressing FCTN REDO, cursoring to the end of the line and continuing input.

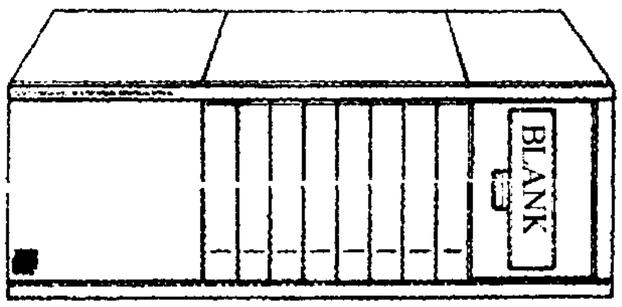
TEX COMP LTD.



WE HAVE RECENTLY ACQUIRED THE REMAINING STOCK FROM TM DIRECT (AKA TRITON). AFTER THE SEMI LEFT MY STORAGE AREA I WAS LEFT WITH A PROBLEM, I CAN'T STORE ALL THE PBOXS AND CARTRIDGES I NOW HAVE. MY LOSS IS YOUR GAIN, TEX- COMP LTD IS HAVING A TRUCK LOAD SALE. LOOK BELOW FOR DRASTIC PRICES REDUCTIONS.



TRUCK LOAD SALE



P-BOXES NEW / RECONDITIONED UNITS IN ORIGINAL TI PACKAGING WITH INTERFACE CARD, POWER CABLE, AND P-BOX INSTRUCTION MANULE FROM TI.

SUPER BUY!

SALE PRICE $35

FOR A LIMITED TIME ONLY!

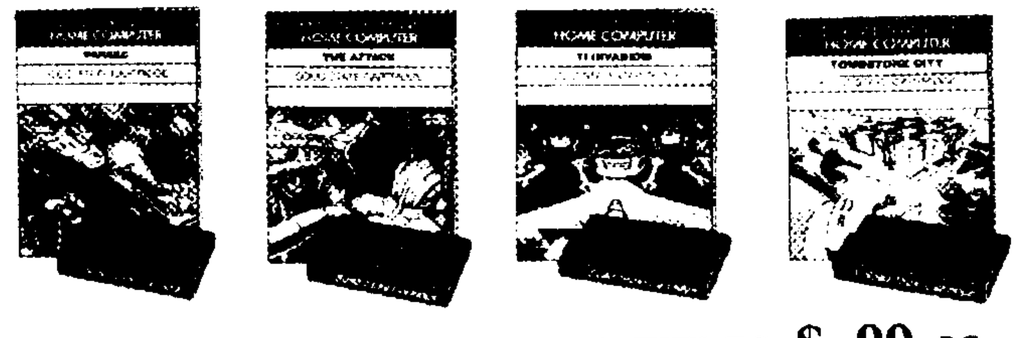
ADD $7 FOR S/H PER P-BOX

MAGIC MEMORY

MAGIC MEMORY belongs in every TI99/4A, give your TI the gift of power. Magic Memory brings many of the average users needs right into one handy cartridge. With features like disk manager functions that rival DM1000, and editor assembler loader functions. It also offers the advanced user easy access to many programs that are used regularly like memory manager, advanced sector editor, debugger, and an editor/assembler cartridge that does not need the editor or assembler files off the disk (a real time saver) all in one convenient cartridge. MAGIC MEMORY comes with a 58 page owners manual that will help you get started and explain all the features contained in the cartridge. **$49.95**

BULK CART WITH MANULES NEVER PACKAGED BY TI

PARSEC ATTACK INVADERS TOMB STONE



ANY OF THESE 4 POPULAR TITLES **$ .99 ea.**

MORE TI PACKAGES FROM THE TRUCK

TEACH YOUR SELF XB	**$.75**
TEACH YOUR SELF BASIC	**$.75**
TAX/INV RCD KEEP	**$.99**
PERSONAL REAL STATE	**$.99**

NEW ORIGINAL TI HARDWARE

CONSOLE POWER PACK	**$4.95**
SS/SD FULL HIGHT DRIVES	**$8.95**

ADD $1.5 PER FOR S/H

SUPER AMS CARD

TAKE YOUR TI TO THE NEXT LEVEL! This memory expansion card breaks the 32K barrier allowing for very large E/A programs or multiple XB programs using the included XB packer. The number of XB programs loadable at one time depends on the memory option ordered. Comes with modified assembler allowing use of all the new features and a multi-page manual on disk.

There are two versions to chose from

128K version	**$85**
256K version (recommended)	**$100**
128K to 256K upgrade	**$25**

ADD $5 EXTRA FOR SHIP / INSURANCE

ORDER BY PHONE

VISA & MASTER CARD EXCEPTED

TEX COMP LTD.
AMERICAS NUMBER ONE TI COMPUTER RETAILER
425 E. ARROW HWY. #732
GLENDORA, CA 91740

(800) 846-3474 FAX (818) 858-2785 **ADD 5% SHIPPING {$5.00 MINIMUM}**

ALL PRICES ARE FOR CASH/CHECK, ADD 3.5% FOR CREDIT CARD ORDERS

COMMENTS

Looking forward to Faire

I'll be in Evanston on Oct. 28 for this year's edition of the Chicago TI Faire. What do I expect? I expect to find camaraderie, some interesting workshops, a selection of used and some new equipment, and some familiar faces. How many familiar faces? I don't think it really matters. Numbers no longer are synonymous with success. Our numbers are dwindling, and there's not a lot we can do about it.

But it doesn't mean that we can't enjoy ourselves when we get together. If anything, we should have even a better time than five or six years ago when the Chicago fair attracted many hundreds of visitors and dozens of exhibitors. Back then, while there was a lot to do, the event was hectic. By the end of the day you were tired and done with crowds. Last year the crowd was gone and you didn't feel driven to see everything and hear everything all at once. You could take your time. You can sit in on a workshop and not feel you were missing something. You could take your time. You could have conversations instead of passing comments.

I enjoyed it and I'm looking forward to more of the same.

THIS MONTH

We've got some long articles this month, so pour yourself a cup of coffee before diving in. Included among these is a piece by Mike Wright that explains how CaDD Electronics broke the protection on the SPAD XIII Flight Simulator so it could be uploaded to PC99. CaDD Electronics is located at 81 Prescott Rd., Raymond, NH 03077-2624.

Also you will notice that the front of the issue is taken up with a really long program by frequent contributor W. Leonard Taffs. WINDOW:6 is an Extended BASIC program to allow the user to view up to six documents at a time.

NEXT MONTH

In fact, we had so many long items this month that a late item we received, a report on the fair in Vienna, Austria, will be printed in the November issue.

—J

FEEDBACK

Author offers support

I would like to thank MICROpendium and Dr. Charles Good for the excellent and thorough TI Workshop review published in the July 1995 issue. As Dr. Good pointed out, I was the author of most parts of TI Workshop, including the manual — in fact, I also designed the two-chip circuit that made it possible to stuff 64K of ROM into the cartridge memory space and I also did the CAD work to generate the artwork for the circuit board used in this and a number of other DataBioTics cartridges.

During a three-year period from 1985 to late 1988, I released the shareware program Superbug II and I released SuperSpace, SuperSpace II, Super 4th and TI Workshop (also known as Magic Memory) as commercial products through DataBioTics. I also assisted with product designs and beta testing of a number of other DataBioTics products. I was especially proud of the TI Workshop accomplishment because this project was extremely innovative (as Dr. Good pointed out, it is the largest module software package ever produced for the 99/4A). However, I was very disappointed with the lack of promotion given the product by DataBioTics and its subsequent non-sales.

After all this time, I am happy to see that some users are still getting benefit out of this product. I have not done any 99/4A work since 1989, but I still have my equipment and several shelves full of technical reference material. Even though the financial reward of this project has been extremely disappointing (I only received a miniscule amount of royalties in 1987-1988 and none since), I will be happy to assist current users of the product in any way I can.

I will still supply Superbug II manuals as indicated in the back of the TI Workshop manual and I will be happy to answer any technical questions I can about TI Workshop or any of the other products that I released through DataBioTics. I will also be glad to research or assist with any other 99/4A technical questions that I might be able to answer.

I can be contacted by e-mail at edohmann@aol.com and am in process of setting up an Internet web site for my company at http://www.ortech-engr.com. I can also be reached by telephone at (713) 480-8904.

Edgar Dohmann
Houston, Texas

Hooked on 99/4A

I still enjoy using my TI99/4A computer with the same vigor, interest and enthusiasm as that day in June 1982 when I purchased it. This same enthusiasm carried over when I first read the magazine devoted to this computer, MICROpendium.

This same interest is shown every month as I await the next issue of MI-

(See Page 6)

COMPETITION COMPUTER

800-471-1600

(Nationwide & Canada ORDERS ONLY)

OUR NEW ADDRESS: 350 MARCELLA WAY MILLBRAE, CALIFORNIA 94030

BUY - SELL - TRADE
HARDWARE - SOFTWARE

Call out toll free number and ask for a Free NEW TI Buyers Guide (2$ value)

MasterCard VISA

M.O.s, Checks & COD

COMPETITON COMPUTER BRINGS BACK THE NAVARONE LINE OF CARTRIDGE SOFTWARE , INCLUDING R.S. SOFTWARE PUBLISHING TITTLES NAVARONE ACQUIRED RIGHTS TO AFTER THE DEMISE OF ROMOX. WE ARE CURRENTLY MANUFACTURING THE FOLLOWING CARTRIDGES:

ANT EATER.....$20
CHICKEN COOP.....$20
CONSOLE WRITER.....$25
DATA BASE MANAGEMENT SYSTEM.$30
(FOR DISK BASED SYSTEMS)
DATA BASE MANAGEMENT ENTRY...$20
(FOR CASSETTE SYSTEMS)
DATA BASE MANAGEMENT SORT.....$20
(UTILITY FOR DBM ENTRY)
DISK FIXER $20
FROG STICKERS.....$20
HEN PECKED.....$20
HOMEWORK HELPER.....$25
HOMEWORK HELPER+.....$30
PAINT 'N PRINT A (GP-100,CP-700).....$30
PAINT 'N PRINT B (GP-550, OKIDATA)$30
PAINT 'N PRINT C (IBM,GEMINI).....$30
PRINCESS AND THE FROG.....$20
ROTOR RAIDERS.....$20
SPEED READING A (TEENS +ADULTS).$25
SPEED READING B (FOR CHILDREN).....$25
SUPER DUPER (A FAST DISK COPIER)..$20
TOPPER.....$20
TYPO-MAN.....$20
TYPO II.....$20

DISK SOFTWARE:

EXTENDED GRAPHICS DISK.....$15
EXTENDED GRAPHICS TAPE.....$15
(ADDS COMMANDS AND PRINTERS FOR PAINT 'N PRINT)
MUSIC EDITOR.....$10
SPRITE EDITOR.....$10
SPEED READING DATA DISKS.....$10
(NOT REQUIRED BUT USEFUL WHEN YOU GET TIRED OF READING THE BUILT-IN READING MATTER IN THE CARTRIDGE.)

WE ALSO HAVE THE FOLLOWING FUNWARE CARTRIDGES:

SCHNOZ-OLA.....$20
ST. NICK.....$20
VIDEO VEGAS.....$20
AMBULANCE.....$20
DRIVING DEMON.....$20
HEN HOUSE.....$20
RABBIT TRAIL.....$20

NAVARONE

SUPER-DUPER

Safeguards masters fast. Use the high-speed Super-Duper program cartridge to duplicate your disks, single and double sided, then lock away your originals. For single drive-systems, a special data-compression routine stores most of your information in expanded memory to reduce disk swapping. Most disks are copied in one or two passes—and Super-Duper works even faster with multiple-drive systems. The program automatically formats blank diskettes before copying and allows you to verify your back-ups byte for byte against the originals.

RARE TI SOFTWARE:

PHD 5066 TI PILOT (REQUIRES P-CODE CARD).....$20
(DISK AND 140 PAGE MANUAL)
PHD 5068 COURSE DESIGNER AUTHORING PACKAGE.$15
(TWO DISK SET AND 54 PAGE MANUAL)

NEW HARDWARE:
DUAL HALF HEIGHT 3.5" DISK DRIVE KIT $70

TI-99/4A™ CD ROM

CD ROM UPDATE: NOW SHIPPING BBS VERSION. THIRD PARTY SOFTWARE LICENCES IN PROCESS OF BEING OBTAINED FOR USE BY TI EMULATORS, PC99 AND SWARTZ'S. BECAUSE THE TI EMULATORS RUN ON THE IBM PLATFORM WE HAVE FOUND SOFTWARE LEGAL DEPARTMENTS NERVOUS ABOUT SUBLICENCING. WATCH THIS SPACE FOR FURTHER NEWS. WE OF COURSE WILL BE MAKING THE NAVARONE SOFTWARE AVAILABLE FOR ALL THE TI EMULATORS.

IMAGINE 650 MEGABYTES OF SOFTWARE WRITTEN FOR THE TI99/4A AND 9640 ON ONE CDROM . THAT IS WHAT YOU WILL HAVE AT THE END OF YOUR SUBSCRIPTION TO THE TI 99/4A CDROM. FOR THE PRICE OF 30 -50 PUBLIC DOMAIN DISKS YOU WILL RECEIVE 1900 DISKS INITIALY AND MANY MORE IN PERIODIC UPDATES UNTIL YOU WILL HAVE 650 MEGABYTES OF TI SOFTWARE. THATS A LOT OF SOFTWARE FOR YOUR COST OF ONLY $100 . SEND YOUR ORDER FORM WITH A CHECK OR MONEY ORDER PAYABLE TO MILLBRAE CD ROM,350 MARCELLA WAY, MILLBRAE, CA 94030. VOICE:(415) 697-1108 .FAX:(415)697-7406

YOUR NAME:_____
ADDRESS:_____
CDROM PLAYER AND PLATFORM INFORMATION:
MY CDROM PLAYER IS MADE BY:_____MODEL#:_____
I HAVE IT CONNECTED TO: IBM/COMPATIBLE___MAC___HORIZON SCSI___
IF YOU CHECKED HORIZON SCSI: IT IS CONNECTED TO:TI99/4A___9640___
I USE THE FOLLOWING OPERATING SYSTEM WITH MY CDROM PLAYER:
WINDOWS___DOS___VERSION:_____(3.1 FOR WINDOWS FOR EXAMPLE)
SYSTEM 7.X___MYARC DOS(MDOS)___VERSION_____ TI99/4A____
PLEASE INCLUDE YOUR CDROM PLAYERS SPEC SHEET WITH YOUR ORDER

CONSOLE WRITER

Turns your console into a word processor without disk-drive memory. With just your console, a printer and this unique cartridge program you can begin word processing today. Features include a full screen text editor that lets you Insert and Delete characters and complete lines. The program is extremely easy to use yet powerful enough to handle most home word processing applications.

HOMEWORK HELPER

Makes homework fun, develops basic computer skills. This new educational program for children 8 years and older, features a built-in 20,000 word checker dictionary to identify spelling errors. Homework Helper is a simple-to-use tool for improving study habits, written work quality, and planning class assignments. Homework Helper contains a word processor which includes standard format for book reports and class projects. DISK DRIVE REQUIRED.

FEEDBACK

(Continued from Page 4)

CROpendium. This same vigor is used to understand and use the many programs and articles printed in MICROpendium

Yes, I have used many different types of computers, programmed in over a dozen languages, and still enjoy returning to the TI99/4A computer. This is why I was disturbed in the August 1995 issue of MICROpendium when you said, "But we do hope to continue it as long as a core group maintains its interest in the TI99/4A. This sounds like you want to fade away because of a lack of interest or maybe rising cost. I can assure you that I read MICROpendium and will continue to even if you change it to a 12-page magazine. Yes, I am hooked on using this old TI99/4A and will continue to do so.

Keep up the great work; we are still out here. Maybe some of us have not figured out how to use TI-Writer yet.

Bryant Krause
Mira Loma, California

Few complaints

I was surprised to see the negative letter from Harry Allston in the August 1995 issue. My experiences with Harry have all been positive, so when he writes something of a less-than-positive nature, I sit up and pay attention. He and I have corresponded over the years about TI-Base issues, and he has written several excellent programs and articles in support of the TI community, so I know Harry Allston is *not* a whiner.

While I can agree with Harry in some areas of his letter, I am hard pressed to complain about what we get each month for $35 ($25 if you are on a fixed income) a year from MICROpendium. John Koloen and Laura Burns may not catch every typo in each issue or churn out picture perfect magazines every time, but the fact that they churn them out at all is worth something to me. Figure it out, folks, with 2,500 subscribers, at $35 each, that's a whopping $87,500 a year before any expenses. Assuming a normal profit margin, or 20-25 percent, that means the two of them make about $10,000 a year each to put the magazine together. You can't buy half a new minivan for $10,000!

When MICROpendium ceases to exist, Bill Gaskill will cease being a supporter of the TI community. Without the communication vehicle MICROpendium provides, there is no community. My hope is that Burns and Koloen are as fanatical in their love of the 99/4A as I am. Keep up the good work, folks!

Bill Gaskill
Grand Junction, Colorado
(Wish we did make $10,000 each from MICROpendium. — Ed.)

Average user left out

I do concur with a point I believe Harry Allston (Feedback, August 1995) was trying to make (albeit his manner being a bit negative).

The point I believe he was trying to make was that perhaps the majority of TI users are being left out (or behind) — and *have been* for some time— by the tendency of TI user group publications that could be termed "too advanced," "too technical" or "too much for the privileged few" (the "*minority* that had other equipment," per Allston,. From my former experience as our newsletter librarian, I found it increasingly discouraging to see so much of this happening, more and more, as time passed. For example, I think very highly of Charles Good, but it has been some time since a Lima newsletter has had anything in it that could be deemed much help to any "average" TI user group member. Perhaps it is time for *another poll* of what's left of the TI community. The writing has been on the wall for some time, we all know, but there are good years left, if we don't blow it.

I continue to staunchly *support you* and not just because you have seen any of my material fit to publish. Thank you!

W. Leonard Taffs
SouthWest 99ers
Tucson, Arizona

Mistakes happen in other publications

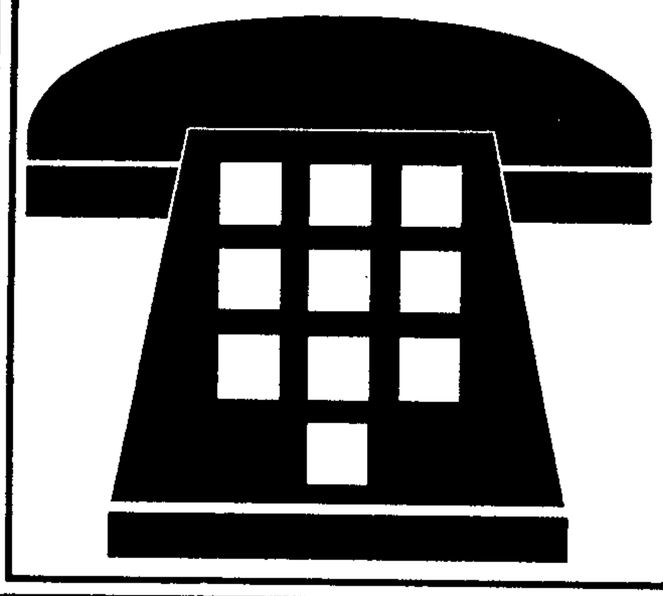
Excuse me! H.A., Reedley, Californ (Feedback, August 1995) complains one misspelled word and promises not to renew his subscription.

If he reads his daily newspaper, in California or any place else in the United States, he will find misprints, misspelling and even wrong captions on wrong pictures.

Forget it — you guys do a fabulous job. Keep going. The TI community loves you.

Ed Mandich
East McKeesport, Pennsylvania

Send your letters and comments to *MICROpendium Feedback*, P.O. Box 1343, Round Rock, TX 78680.



Want to talk to someone at MICROpendium?

You'll need to call between the hours of 9 a.m. and noon Saturdays. If you call at other times, you will probably get an answering machine. But don't let that bother you. We listen to the answering machine at least once a day and return calls as soon as possible, usually that day.

Call us at
512-255-1512

Extended BASIC

View up to six documents at once with WINDOW:1-6

By W. LEONARD TAFFS
SouthWest 99ers

WINDOW:1-6 is an Extended BASIC program that lets users view up to six documents simultaneously on the screen. This capability can be useful to anyone who wants to compare text files of any kind. The program is listed below, but full documentation is too voluminous for publication. These files, along with the program, will be included on the monthly MICROpendium disk for October.

WINDOW:1-6 Vs.1.2 is an updated version of WINDOW:1-6. These docs and V. 1.2 program supersede the program segment listings and document extracts appearing in the Feedforth columns of SouthWest 99ers newsletters (May through September 1995).

This program requires at least one disk drive and 32K memory expansion and Extended BASIC. A printer is desirable.

You can run the WINDOW:1-6 program without the docs, but there are several escapes or extra key-press options that could not be included in the program because lack of space in memory to show them on user screen.

This program eats up a lot of stack space, especially when reading long files. It can manage six D/V 80 files at a time pretty well if they are of moderate size (moderate meaning 16 to 25 sectors each). A ballpark guesstimate is that the program reads between 700 and 1000 lines of text (depending upon length of individual record lines) before the stack is depleted to the point the program stops. An "ON ERROR" direction (line 430) returns the program to the opening title screen without clearing the screen. The program must be re-run at this point regardless of screen prompt.

Single disk drive users and non-RAMdisk multiple drive users will need to do a CALL FILES if they wish to open more than 3 files. (Allow 1 file for catalog option use in program.) Unfortunately, the more files you include in the CALL FILES, the more stack space is sacrificed, limiting the length of files that can be read before stack is used up. A CALL FILES(7) will cost you 2072 stack bytes.

This program eats up a lot of stack space, especially when reading long files. It can manage six D/V 80 files at a time pretty well if they are of moderate size (moderate meaning 16 to 25 sectors each).

Pre-Title Screen prompts have been modified in the program to distinguish between users with RAMdisk multiple drives and non-RAMdisk multiple drives. RAMdisk (Horizon 4000) users will be able to open six files AND use catalog option without having to do a CALL FILES.

If program fails to accept your drive numbers (3 and 4 are excluded in the ACCEPT AT validation strips), then edit these ACCEPT AT lines to validate drives you need. As is, this program validates drives 0, 1, 2, 5, 6, 7, 8, 9, A, B, C, and D (lines 2280 and 2760). 0 is used for escape.

Omitted from the docs was mention of the fact that you have another quick exit from the main program in the "L" (open a file) option. Enter "QUIT" as the filename input to leave the program. If the program is ended without the user closing files, it will print (to screen) the last record, record number, and filename of any INPUT files opened.

At the bottom of page 8 of the docs there is mention of using "BACK" as a filename in the "L" option which will take you back to opening prompts of the program. However, this will not clear variables of the program as you continue. Before using BACK, the user should close all open files. If you use BACK and do not close any open files first, it will create an error when you enter opening prompts for filenames. This error will be evident when the prompt "Open HOW MANY FILES.." keeps returning after you attempt to open more files. In this case, enter a "7" to return to main program to use the "M" option to close each of any open files. Then you can opt to open individual files ("L" option) or use the "BACK" again to open them.

The program is almost 100 percent crash-free if directions are followed. The only mistake that surely will crash the program, if you do not respond correctly to screen instructions, is explained on page 9 of the docs.

If you have any problems with this program, please let me know. If this reaches anyone who has copied all the listings from SouthWest 99ers newsletters, write me for additional lines and amended lines to add to the original version. Or you can send me a SASE for the new program listing, or send a SAS Diskmailer and blank disk (or 2 disks if you require 360 format) for a free copy of all files. If you write, please do not use SW99ers mailing address. This may delay your correspondence reaching me.

Taffs can be reached at 4124 E. First St., Tucson, AZ 85711-1006.

WINDOW:1-6

```
1 ! [WINDOW:1-6] Vs1.2 42595
inspired by Prog from K-TOWN
99er p.2 9409 by TOM MORAN-
```

(See Page 8)

WINDOW:1-6 —

(Continued from Page 7)

```
VAST NEWS. FILE READER by W.
L.Taffs, SW99ers, Tucson AZ.
 !218
2 !!131
3 !   Opens 6 DV/80 files
      and windows records.
      Files 1-3 windows 1-3
      Files 4-6 windows 4-6
      Record by Record Step
!225
4 !!131
5 ! MEMO Output File Option
    Printer Option
    Opens/Closes Input Files
    Selected records printed
 !073
6 !!131
7 ! IF NOT USING A RAMDISK
    You must do a CALL FILES
    (n) for more than 3 fils
    (7 Max). CATALOG option
    will require 1 of these.
 !117
8 !!131
40 CLF$="type NEW" :: CLG$="
type CALL FILES(n)" :: CLH$=
"type NEW" :: CLI$="type RUN
 ""DSKn.WINDOW:1-6""" :: CLJ
$="maximum call files (in th
is program) = 7." !059
45 CLK$="Using CATALOG takes
 one file" !026
50 CALL BLUE :: CALL CLEAR :
: CALL SCREEN(8):: CALL CHAR
(128,"FFFF"):: CALL CHAR(129
,"8080808080808080010101010101
0101010000000000000FF")!108
51 DISPLAY AT(15,2):"SINGLE
DRIVE users ENTER ""1""": :"
MULTIPLE Drive Use ENTER ""0
""" :: ACCEPT AT(20,14)VALID
ATE("01")SIZE(-1):SDO !125
52 IF SDO THEN GOSUB 4100 !1
26
53 IF SDO=0 THEN PRINT :"ARE
 YOUR DRIVES RAMDISKS?": : :
: INPUT "Enter 1=YES or 0=NO
 ":RMD :: IF RMD THEN 55 ELS
E GOSUB 4100 !042
55 ND$="Please Send Comments
 or any Questions about this
 programto W.Leonard Taffs,
SW99ers,4124 E.First St., TU
CSON,Az.85711-1006. Thanks!"
 !112
60 CALL CLEAR :: CALL HCHAR(
1,2,131,30):: CALL HCHAR(15,
2,131,30):: CALL VCHAR(2,1,1
30,14):: CALL VCHAR(2,32,129
,14)!202
70 T$="WINDOW:1-6    Version
 1.2" :: T2$="By W.LEONARD T
AFFS, SW99ERS" :: T3$="Tucso
n, Arizona 04/95" :: T6$="Ca
n interchange viewing them"
!224
80 T4$="inspired by Tom Mora
n, VAST    via K-Town 99er 9
/94" :: T5$="6 D/V 80 Files
can be opened" !112
90 ST1$="press respective wi
ndow #" :: ST2$="to bring fi
le into view." :: ST3$="use
shift/# for resident rec" !0
45
100 DISPLAY AT(3,2):T$ :: DI
SPLAY AT(5,1):T2$ :: DISPLAY
 AT(6,4):T3$ :: DISPLAY AT(8
,1):T4$ :: DISPLAY AT(11,1):
T5$: :T6$ !085
110 DISPLAY AT(21,1):"OPEN H
OW MANY FILES? (1-6)" :: IF
SDO OR RMD=0 THEN DISPLAY AT
(23,1):"CALL FILES NECESSARY
 FOR >3!" !115
120 ACCEPT AT(21,28)SIZE(-1)
:HM :: IF HM<0 OR HM>7 THEN
120 :: IF HM=0 THEN GOSUB 22
60 ELSE IF HM=7 THEN HM=0 ::
 GOTO 1540 ELSE 130 !239
130 DISPLAY AT(21,1):"SCAN O
NE FILE AT A TIME (1)" :: DI
SPLAY AT(23,1):"OR ALL SIMUL
TANEOUSLY?  (0)" :: ACCEPT A
T(23,26)SIZE(-1):SL !217
140 IF SL=3 THEN SL=0 :: FR=
1 !181
143 IF FR THEN PRINT :: INPU
T "USE SEARCH? (0=NO 1=YES)
":SCH !011
146 IF SCH THEN PRINT :: INP
UT "SEARCH for: ":SCH$ !195
150 IF SL=0 THEN J=1 :: DISP
LAY AT(19,1):RPT$(" ",140)::
 DISPLAY AT(22,2):"SET VIEW
DELAY TIME:" !056
160 IF SL=0 THEN ACCEPT AT(2
2,23)VALIDATE("0123456789")S
IZE(-4):TM :: IF TM>4000 THE
N 160 !097
170 DISPLAY AT(19,2):RPT$("-
",84):: DISPLAY AT(19,2):"En
ter Date (00/00/00)" :: ACCE
PT AT(19,14)VALIDATE("012345
67890/")SIZE(-8):DT$ !003
180 CALL CLEAR :: IF HM>=1 T
HEN INPUT "FIRST FILE NAME?
":F1$ :: PRINT :: INPUT "ON
DSK #: ":D1$ :: PRINT !097
190 F1$="DSK"&D1$&"."&F1$ ::
 F1=1 :: IF HM=1 THEN GOTO 3
40 !175
200 IF HM>=2 THEN INPUT "SEC
OND FILE NAME? ":F2$ :: PRIN
T :: INPUT "ON DSK #: ":D2$
:: PRINT !070
210 F2$="DSK"&D2$&"."&F2$ ::
 F2=2 :: IF HM=2 THEN 340 !0
47
230 IF HM>=3 THEN INPUT "THI
RD FILE NAME? ":F3$ :: PRINT
 :: INPUT "ON DSK #: ":D3$ :
: PRINT !007
240 F3$="DSK"&D3$&"."&F3$ ::
 F3=3 :: IF HM=3 THEN GOTO 3
40 !187
260 IF SDO THEN 340 ELSE IF
HM>=4 THEN INPUT "FOURTH FIL
E NAME? ":F4$ :: PRINT :: IN
PUT "ON DSK #: ":D4$ :: PRIN
T !033
270 F4$="DSK"&D4$&"."&F4$ ::
 F4=4 :: IF HM=4 THEN 340 !0
59
290 IF HM>=5 THEN INPUT "FIF
TH FILE NAME? ":F5$ :: PRINT
 :: INPUT "ON DSK #: ":D5$ :
: PRINT !003
300 F5$="DSK"&D5$&"."&F5$ ::
 F5=5 :: IF HM=5 THEN 340 !0
65
320 IF HM>=6 THEN INPUT "SIX
TH FILE NAME? ":F6$ :: PRINT
 :: INPUT "ON DSK #: ":D6$ :
: PRINT !037
330 F6$="DSK"&D6$&"."&F6$ ::
 F6=6 !033
340 CALL CLEAR :: DISPLAY AT
(6,1):"To Open these files:"
 :: DISPLAY AT(9,5):F1;F1$ :
: IF F2 THEN DISPLAY AT(11,5
):F2;F2$ !223
350 IF F3 THEN DISPLAY AT(13
```

(See Page 10)

MICROpendium

DISK SALE

If you've been waiting for a sale on MICROpendium program disks, this is it! For a limited time (through Nov.1, 1995) Series 1-8 disks are available for a special price. (Series 8 disks are mailed monthly starting with the September 1995 edition, programs from April 1995 through September 1995 will be mailed as soon as the order is placed.)

MICROpendium disks

SERIES #	REGULAR PRICE	SALE PRICE	YOU SAVE	DISCOUNT
Series 1 (Apr. '88-Mar. '89)	$25.00	**$15.00**	*$10.00*	*40%*
Series 2 (Apr. '89-Mar. '90)	$25.00	**$15.00**	*$10.00*	*40%*
Series 3 (Apr. '90-Mar. '91)	$25.00	**$15.00**	*$10.00*	*40%*
Series 4 (Apr. '91-Mar. '92)	$25.00	**$15.00**	*$10.00*	*40%*
Series 5 (Apr. 92-Mar. 93)	$25.00	**$15.00**	*$10.00*	*40%*
Series 6 (Apr. 93-Mar. 94)	$25.00	**$15.00**	*$10.00*	40%
Series 7 (Apr. 94-Mar. 95)	$2500	**$15.00**	*$10.00*	40%
Series 8 (Apr. 95-Mar. 96)	$40.00	$25.00	$15.00	38%

(Circle the items you want to order)

ACT NOW — THIS SALE ENDS NOV. 1, 1995

Customer information

Name _____

Address _____

City _____

State _____

ZIP _____

Please circle the items above and return this entire page (or a copy of it) with a check or money order in payment.

Total Remitted _____

(U.S. Funds only; Texas residents add 7.75% sales tax)

Disk shipping information

Postage is included for any disk sales to U.S. addresses. **Canadian delivery**: add $2.00 for each series of disks for airmail delivery, $1.50 for surface. **Overseas delivery**: add $3.50 for each series of disks for airmail delivery; add $2.00 for each series for surface

FOR CREDIT CARD ORDERS

Credit Card No. _____

Credit Card: MC Visa (Circle One) **Exp. Date** _____

Signature _____

(credit card orders only; credit card orders add 5%)

WINDOW:1-6 —

(Continued from Page 8)

```
,5):F3;F3$ !010
360 IF F4 THEN DISPLAY AT(15
,5):F4;F4$ !015
370 IF F5 THEN DISPLAY AT(17
,5):F5;F5$ !020
380 IF F6 THEN DISPLAY AT(19
,5):F6;F6$ !025
390 DISPLAY AT(23,10):"O.K.?
 (Y/N)" :: ACCEPT AT(23,22)V
ALIDATE("NYny")SIZE(-1):K$ !
158
400 IF K$<>"Y" AND K$<>"y" T
HEN CALL CLEAR :: F1,F2,F3,F
4,F5,F6,HM,SL=0 :: F1$,F2$,F
3$,F4$,F5$,F6$="" :: GOTO 10
0 !149
410 CALL CLEAR :: DISPLAY AT
(15,6):"i'm opening files!"
:: ST=1 !004
430 ON ERROR 1440 :: IF F1 T
HEN OPEN #1:F1$,INPUT !006
440 IF F2 THEN OPEN #2:F2$,I
NPUT !217
450 IF F3 THEN OPEN #3:F3$,I
NPUT !220
460 IF F4 THEN OPEN #4:F4$,I
NPUT !223
470 IF F5 THEN OPEN #5:F5$,I
NPUT !226
480 IF F6 THEN OPEN #6:F6$,I
NPUT !229
500 CALL CLEAR :: CALL HCHAR
(1,2,128,30):: CALL HCHAR(8,
2,128,30):: CALL VCHAR(1,1,1
30,7):: CALL VCHAR(1,32,129,
7):: IF FV THEN RETURN ! W1
!076
505 IF (F1=0)*(F2=0)*(F3=0)*
(F4=0)*(F5=0)*(F6=0)THEN DIS
PLAY AT(3,2):S19$ !243
510 CALL HCHAR(9,2,128,30)::
 CALL HCHAR(16,2,128,30):: C
ALL VCHAR(9,1,130,7):: CALL
VCHAR(9,32,129,7)! WINDOW 2
!009
520 CALL HCHAR(17,2,128,30):
: CALL HCHAR(24,2,128,30)::
CALL VCHAR(17,1,130,7):: CAL
L VCHAR(17,32,129,7)! WINDOW
 3 !153
530 !!131
540 IF (HM=7)+(ST=1)THEN DIS
PLAY AT(3,2):ST1$ :: DISPLAY
 AT(4,3):ST2$ :: DISPLAY AT(
5,3):ST3$ :: IF HM=7 THEN HM
=0 !240
550 IF K=77 OR K=80 THEN K=0
 :: GOTO 1060 ELSE IF V THEN
 V=0 :: GOTO 1060 !118
560 ST=0 :: IF J THEN G1,G2,
G3=1 ELSE IF J2 THEN G4,G5,G
6=1 !170
570 IF F1 THEN IF EOF(1)THEN
 F1,C1,FR,G1,SCH=0 :: SCH$,F
1$="" :: CLOSE #1 :: GOSUB 3
450 :: GOTO 590 !086
580 IF G1 THEN IF F1 THEN C1
=C1+1 :: LINPUT #1:A1$ :: GO
SUB 800 :: IF J THEN 590 ELS
E GOTO 720 !214
590 IF F2 THEN IF EOF(2)THEN
 F2,C2,G2=0 :: F2$="" :: CLO
SE #2 :: GOSUB 3450 :: GOTO
620 !234
600 IF J THEN GOSUB 3370 !00
5
610 IF G2 THEN IF F2 THEN C2
=C2+1 :: LINPUT #2:A2$ :: GO
SUB 850 :: IF J THEN 620 ELS
E GOTO 720 !044
620 IF F3 THEN IF EOF(3)THEN
 F3,C3,G3=0 :: CLOSE #3 :: G
OSUB 3450 :: GOSUB 3310 :: G
OTO 650 !055
640 IF G3 THEN IF F3 THEN C3
=C3+1 :: LINPUT #3:A3$ :: GO
SUB 900 :: IF J THEN GOSUB 3
170 ELSE 720 !101
650 IF F4 THEN IF EOF(4)THEN
 F4,C4,G4=0 :: CLOSE #4 :: G
OSUB 3450 :: GOTO 670 !133
660 IF G4 THEN IF F4 THEN C4
=C4+1 :: LINPUT #4:A4$ :: GO
SUB 940 :: IF J2 THEN 670 EL
SE 720 !112
670 IF F5 THEN IF EOF(5)THEN
 F5,C5,G5=0 :: CLOSE #5 :: G
OSUB 3450 :: GOTO 690 !159
680 IF G5 THEN IF F5 THEN C5
=C5+1 :: LINPUT #5:A5$ :: GO
SUB 980 :: IF J2 THEN 690 EL
SE 720 !178
690 IF F6 THEN IF EOF(6)THEN
 F6,C6,G6=0 :: CLOSE #6 :: G
OSUB 3450 !032
700 IF G6 THEN IF F6 THEN C6
=C6+1 :: LINPUT #6:A6$ :: GO
SUB 1020 :: IF J2 THEN GOSUB
 3170 :: GOTO 650 ELSE 720 !
126
710 IF AFC THEN DISPLAY AT(2
1,1):"all files closed!":S19
$:"ready to re-run..." :: RT
=1 :: GOSUB 3340 :: GOTO 135
0 ELSE 720 !205
720 IF J OR J2 THEN HLP=1 ::
 GOSUB 3130 :: GOTO 740 !031
730 G1,G2,G3,G4,G5,G6=0 :: C
ALL KEY(0,K,S):: IF S<1 THEN
 730 :: DISPLAY AT(1,28):CHR
$(K)!094
740 IF RT THEN F1,F2,F3,F4,F
5,F6,PR,C1,C2,C3,C4,C5,C6=0
:: IF RT THEN 2 !094
750 !!131
760 FOR I=9 TO 12 :: CALL CO
LOR(I,16,1):: NEXT I :: CALL
 SCREEN(3)!156
790 GOTO 1060 !119
800 DISPLAY AT(3,2):RPT$(" "
,112)!166
810 DISPLAY AT(2,2):SEG$(F1$
,4,12);C1 :: DISPLAY AT(4,2)
:SEG$(A1$,1,27):: DISPLAY AT
(5,2):SEG$(A1$,28,27):: DI
LAY AT(6,2):SEG$(A1$,55,26,
088
820 DISPLAY AT(2,28):"1" !14
3
822 IF FR THEN CALL KEY(0,K,
S):: IF S<>1 THEN 824 !013
823 IF K=80 THEN GOSUB 2110
:: RETURN ELSE IF K=69 THEN
CLOSE #1 :: C1,F1,G1,FR,K,SC
H=0 :: F1$,SCH$="" :: GOTO 5
00 !150
824 IF SCH THEN IF POS(A1$,S
CH$,1)THEN CALL KEY(0,K,S)::
 IF S<1 THEN 824 !005
825 IF K=80 THEN GOSUB 2110
!005
826 IF K=89 THEN K=0 :: GOTO
 1740 ELSE IF FR THEN 570 !1
11
830 IF J THEN GOSUB 3430 ::
GOTO 590 ELSE RETURN ELSE IF
 HLP THEN 850 ELSE RETURN !0
43
850 DISPLAY AT(11,2):RPT$("
",112)!214
860 DISPLAY AT(10,2):SEG$(I
$,4,12);C2 :: DISPLAY AT(12,
2):SEG$(A2$,1,27):: DISPLAY
```

(See Page 11)

WINDOW:1-6 —

(Continued from Page 10)

```
AT(13,2):SEG$(A2$,28,27):: D
ISPLAY AT(14,2):SEG$(A2$,55,
26)!029
870 DISPLAY AT(10,28):"2" !1
92
880 IF J THEN GOTO 620 ELSE
RETURN ELSE IF HLP THEN 900
ELSE RETURN !054
900 DISPLAY AT(19,2):RPT$("
",112)!222
910 DISPLAY AT(18,2):SEG$(F3
$,4,12);C3 :: DISPLAY AT(20,
2):SEG$(A3$,1,27):: DISPLAY
AT(21,2):SEG$(A3$,28,27):: D
ISPLAY AT(22,2):SEG$(A3$,55,
26)!039
920 DISPLAY AT(18,28):"3" !2
01
930 IF J THEN GOSUB 3170 ::
GOTO 560 ELSE IF HLP THEN 94
0 ELSE RETURN !089
940 DISPLAY AT(3,2):RPT$(" "
,112)!166
950 DISPLAY AT(2,2):SEG$(F4$
,12);C4 :: DISPLAY AT(4,2)
:SEG$(A4$,1,27):: DISPLAY AT
(5,2):SEG$(A4$,28,27):: DISP
LAY AT(6,2):SEG$(A4$,55,26)!
103
960 DISPLAY AT(2,28):"4" !14
6
970 IF J2 THEN 670 ELSE IF H
LP THEN 980 ELSE RETURN !091
980 DISPLAY AT(11,2):RPT$("
",112)!214
990 DISPLAY AT(10,2):SEG$(F5
$,4,12);C5 :: DISPLAY AT(12,
2):SEG$(A5$,1,27):: DISPLAY
AT(13,2):SEG$(A5$,28,27):: D
ISPLAY AT(14,2):SEG$(A5$,55,
26)!044
1000 DISPLAY AT(10,28):"5" !
195
1010 IF J2 THEN 690 ELSE IF
HLP THEN 1020 ELSE RETURN !1
51
1020 DISPLAY AT(19,2):RPT$("
",112)!222
1030 DISPLAY AT(18,2):SEG$(F
6$,4,12);C6 :: DISPLAY AT(20
,2):SEG$(A6$,1,27):: DISPLAY
 AT(21,2):SEG$(A6$,28,27)::
DISPLAY AT(22,2):SEG$(A6$,55
,26)!054
1040 DISPLAY AT(18,28):"6" !
204
1050 IF J2 THEN GOSUB 3170 :
: RETURN ELSE HLP=0 :: RETUR
N !158
1060 ! ** CALL KEYS ** !003
1070 IF K=87 THEN GOTO 500 !
186
1075 IF K=68 THEN GOSUB 4300
 !161
1080 IF K=66 THEN SL,PT=0 ::
 BB=1 :: GOTO 3170 !029
1090 IF K=80 THEN GOSUB 2110
 !005
1100 IF K=67 THEN GOSUB 2260
 !160
1110 IF K=49 THEN IF F1 THEN
 G1=1 :: GOTO 570 !092
1120 IF K=33 THEN IF F1 THEN
 GOSUB 800 !139
1130 IF K=50 THEN IF F2 THEN
 G2=1 :: GOTO 590 !106
1140 IF K=64 THEN IF F2 THEN
 GOSUB 850 !194
1150 IF K=51 THEN IF F3 THEN
 G3=1 :: GOTO 620 !139
1160 IF K=35 THEN IF F3 THEN
 GOSUB 900 !243
1170 IF K=52 THEN IF F4 THEN
 G4=1 :: GOTO 650 !172
1180 IF K=36 THEN IF F4 THEN
 GOSUB 940 !029
1190 IF K=53 THEN IF F5 THEN
 G5=1 :: GOTO 670 !195
1200 IF K=37 THEN IF F5 THEN
 GOSUB 980 !071
1210 IF K=54 THEN IF F6 THEN
 G6=1 :: GOTO 690 !218
1220 IF K=94 THEN IF F6 THEN
 GOSUB 1020 !115
1230 IF K=76 THEN GOTO 2670
!059
1240 IF K=77 THEN GOSUB 2900
 !036
1250 IF K=81 THEN 1310 !092
1260 IF K=57 THEN GOSUB 1740
 !149
1270 IF K=86 THEN V=1 :: GOS
UB 3020 !044
1280 IF K=72 THEN GOSUB 1540
 !202
1290 IF J THEN 550 ELSE 720
!139
1300 ! ** END (Q) CALLED * !
191
1310 DISPLAY AT(21,1):"Q has
 been activated to end.sure
you want to end? (Y/N)" :: A
CCEPT AT(23,25)VALIDATE("NYn
y")SIZE(-1):YN$ !052
1320 CALL CLEAR :: IF YN$<>"
Y" AND YN$<>"y" THEN 500 ELS
E DISPLAY AT(21,1):RPT$(" ",
56)!035
1330 CALL CLEAR :: DISPLAY A
T(21,1):"run again? (Y/N)" :
: ACCEPT AT(21,18)VALIDATE("
NYny")SIZE(-1):RA$ :: IF RA$
<>"Y" AND RA$<>"y" THEN 1350
 !224
1340 CALL CLEAR :: CALL SCRE
EN(14):: DISPLAY AT(15,1):"P
ROGRAM NOW RE-RUNNING.....":
 :"please give me 25 seconds
!" :: RUN !138
1350 CALL CLEAR :: CALL SCRE
EN(10):: PRINT "  PROGRAM US
ER TERMINATED." !007
1360 GOSUB 4500 !245
1370 IF RT THEN RT=0 :: GOTO
 1 !009
1430 PRINT :TAB(4);"ALL FILE
S NOW CLOSED!": :"    PROGRA
M TERMINATED." :: IF CA THEN
 CA=0 :: GOSUB 3340 :: GOTO
1 ELSE GOTO 1330 !022
1440 ! * ON ERR window files
 * !200
1450 ON ERROR 1520 :: IF F1
THEN CLOSE #1 !194
1460 ON ERROR 1520 :: IF F2
THEN CLOSE #2 !196
1470 ON ERROR 1520 :: IF F3
THEN CLOSE #3 !198
1480 ON ERROR 1520 :: IF F4
THEN CLOSE #4 !200
1490 ON ERROR 1520 :: IF F5
THEN CLOSE #5 !202
1500 ON ERROR 1520 :: IF F6
THEN CLOSE #6 !204
1510 ON ERROR 1520 :: IF F7
THEN CLOSE #7 !206
1520 IF AFC THEN AFC=0 ELSE
F1$,F2$,F3$,F4$,F5$,F6$=" "
:: F1,F2,F3,F4,F5,F6=0 :: GO
TO 110 !166
1530 GOTO 720 !033
1540 ! ** K-PRESS HELP ** !2
17
```

(See Page 12)

WINDOW:1-6 —

(Continued from Page 11)

```
1550 CALL CLEAR :: HLP=1 ::
IF HM=7 THEN HM=0 !068
1560 S2$="* USE 1/4 TO ADV T
OP WINDOW" :: S3$="* USE 2/5
 TO ADV MID WINDOW" :: S4$="
* USE 3/6 TO ADV BOT WINDOW"
 !221
1570 S5$="* USE SHFT/WNDW #
TO RETURN" :: S6$="       FO
R RESIDENT RECORD" :: S7$="*
 USE H TO SEE HELP SCREEN" !
150
1580 S8$="* USE P TO PRINT A
 RECORD" :: S9$="* USE L TO
OPEN A FILE" :: S19$="    *
no files open! *" !223
1590 S20$="* USE V TO CHECK
FILE STATS" :: S1$="keypress
 help: mem status" :: IF F14
 THEN S1$=S1$&S18$ !193
1600 S10$="* USE M TO CLOSE
A FILE" :: S17$="any key to
get back to prog" !179
1610 S11$="* USE C TO CATALO
G A DISK" :: S12$="* USE Q T
O EXIT PROGRAM" :: S13$="* U
SE 9 TO OPEN MEMO FILE" !190
1620 S14$="* USE 9 TO RETURN
 TO MEMO" :: S15$="* USE R T
O LEAVE MEMO FILE" :: S16$="
* USE 7 TO CLOSE MEMO FILE"
:: S18$=" op" !192
1630 S21$="* USE W TO CLEAR
WINDOWS" :: S22$="* ALL WIND
OW NOW TAKEN *" !192
1640 IF J THEN 1700 !241
1650 DISPLAY AT(1,2):S1$;F7
:: DISPLAY AT(3,2):S2$ :: DI
SPLAY AT(4,2):S3$ :: DISPLAY
 AT(5,2):S4$ :: DISPLAY AT(6
,2):S5$ !101
1660 DISPLAY AT(7,2):S6$ ::
DISPLAY AT(8,2):S21$ !206
1670 DISPLAY AT(10,2):S7$ ::
 DISPLAY AT(11,2):S11$ :: DI
SPLAY AT(12,2):S9$ :: DISPLA
Y AT(13,2):S10$ !207
1680 DISPLAY AT(14,2):S8$ ::
 DISPLAY AT(15,2):S20$ :: DI
SPLAY AT(16,2):S12$ :: DISPL
AY AT(18,2):S13$ :: DISPLAY
AT(19,2):S14$ !002
1690 DISPLAY AT(20,2):S15$ :
: DISPLAY AT(21,2):S16$ :: D
ISPLAY AT(23,2):S17$ !076
1700 !!131
1710 !!131
1720 CALL KEY(0,K,S):: IF S<
1 THEN 1720 !004
1730 IF CM THEN 1980 ELSE ST
=1 :: GOSUB 500 :: RETURN !0
01
1740 ! ** CREATE MEMO ** !14
1
1750 CALL CLEAR :: CM=1 !155
1755 IF SDO THEN GOSUB 4400
:: INPUT "* press <ENTER> to
 continue ":K$ :: CALL CLEAR
 !080
1760 CALL HCHAR(1,2,131,30):
: CALL HCHAR(20,2,131,30)::
CALL VCHAR(2,1,130,19):: CAL
L VCHAR(2,32,129,19)!125
1770 CALL CHARSET :: CALL BL
UE :: DISPLAY AT(3,1):"CREAT
E MEMO OPTION. CAREFUL!" ::
DISPLAY AT(5,2):"File Errors
 may cause loss" !016
1780 DISPLAY AT(6,10):"of Da
ta." :: DISPLAY AT(7,1):"Cur
rent Memo STATUS is:";F7 ::
DISPLAY AT(8,1):"last note:"
: :"""";MEM$;"""" !220
1790 ! IF SDO THEN GOSUB 440
0 :: GOTO 1830 !232
1800 DISPLAY AT(16,1):"USE "
"P"" TO PRINT W/O SAVE" !199
1810 DISPLAY AT(18,1):"USE "
"C"" To Continue" :: DISPLAY
 AT(22,1):"Enter Choice:" !2
10
1820 DISPLAY AT(22,28):CHR$(
DSP)!243
1830 CALL KEY(3,K,S):: IF S<
>1 THEN 1830 !054
1840 IF K=72 THEN DSP=K :: G
OSUB 1540 !060
1850 IF K=69 THEN DSP=K :: F
V,CM=0 :: ST=1 :: GOSUB 500
:: GOTO 720 !201
1855 IF K=70 THEN FV=1 !194
1860 IF K=80 THEN 1980 !251
1890 IF K<>67 THEN DSP=K ::
GOTO 1820 !025
1900 IF F7 THEN 1980 :: CALL
 CLEAR :: INPUT "Enter MEMO
Filename:          ":F7$ ::
 IF LEN(F7$)>10 THEN 1900 ::
 PRINT !229
1905 IF F7$="EXIT" THEN F
SF$="" :: K=69 :: GOTO 1850
!004
1910 DISPLAY AT(23,1):"To Ds
k #? " :: ACCEPT AT(23,11)VA
LIDATE("01256789ABCDE")SIZE(
-1):D$ !254
1915 F7$="DSK"&D$&"."&F7$ ::
 PRINT :F7$: :"O.K.? (Y/N)"
:: INPUT "":YN$ :: IF YN$<>"
Y" AND YN$<>"y" THEN 1900 !0
92
1920 PRINT :: INPUT "ABSOLUT
ELY SURE? (Y/N) ":K$ :: IF K
$<>"Y" AND K$<>"y" THEN 1740
 !005
1930 CALL CLEAR :: CALL SCRE
EN(14):: DISPLAY AT(15,1):"O
PENING OUTPUT FILE....."
: :"     ";F7$ !058
1940 DISPLAY AT(20,5):"Press
 <ENTER> to Proceed....": :"
 use ""A"" for emergency abo
rt" !019
1950 CALL KEY(0,K,S):: IF S<
>1 THEN 1950 !171
1960 IF K=65 THEN F7$,D7$=""
 :: F7,C7=0 :: CALL CLEAR ::
 GOTO 1740 !195
1970 OPEN #7:F7$,OUTPUT :: F
7=1 :: SF$=F7$ !241
1980 CALL CLEAR :: CALL HCHA
R(10,3,131,29):: CALL HCHAR(
20,2,131,30):: CALL VCHAR(11
,2,130,10):: CALL VCHAR(11,3
2,129,10)!091
1990 DISPLAY AT(9,1):"memo s
tat:";F7;SF$ !154
1995 IF FV THEN GOSUB 500 ::
 GOSUB 4000 :: DISPLAY AT(2,
1):GU$: :GV$ :: FV=0 !052
1996 GW$=GU$ :: GX$=GV$ :: G
Y$=GW$&" "&GX$ :: GU$,GV$=""
 !032
1997 DISPLAY AT(2,26):GV !04
9
2000 DISPLAY AT(12,2):"USE "
"S"" TO SAVE EACH ENTRY" ::
DISPLAY AT(14,2):"USE ""R""
TO RETURN TO FILES" !185
2010 DISPLAY AT(16,2):"USE "
"7"" TO CLOSE MEMO FILE" ::
DISPLAY AT(18,2):"USE ""P""
FOR PRINTER ONLY":"     set
```

(See Page 13)

WINDOW:1-6 —

(Continued from Page 12)

```
printer on line" !144
2020 DISPLAY AT(22,1):"* ent
er DATA (or press ENTER for
NULL) Then press OPTION" ::
LINPUT "":MEM$ :: IF MEM$="/
/" THEN MEM$=GY$ !242
2030 CALL KEY(0,K,S):: IF S<
1 THEN 2030 !059
2035 IF K=66 THEN 1740 !014
2040 IF K=82 THEN 2080 !098
2050 IF K=83 OR K=115 THEN I
F MEM$<>"" THEN C7=C7+1 :: O
N ERROR 1980 :: PRINT #7:MEM
$ !162
2060 IF K=55 THEN 2090 !108
2070 IF K=80 OR C7 THEN PR=1
 :: GOSUB 2110 !021
2080 IF K=82 THEN PR,CM=0 ::
 ST=1 :: GOSUB 500 :: GOTO 7
20 !088
2090 ON ERROR 1980 :: PRINT
#7:"EOF "&DT$&" "&F7$ :: CLO
SE #7 :: F7$="" :: F7,C7=0 :
: ST=1 :: GOSUB 500 :: GOTO
20 :: RETURN !205
2100 GOTO 1980 !018
2110 ! ** ANY PRINT ** !032
2120 OPEN #9:"PIO",VARIABLE
96 :: IF PR OR C7 THEN 2230
!247
2125 IF SCH THEN KR=1 :: GOT
O 2160 !176
2130 DISPLAY AT(23,1):"print
 which window? (1-7)" :: ACC
EPT AT(23,28)VALIDATE("01234
567")SIZE(-1):KR :: DISPLAY
AT(23,1):RPT$(" ",28)!209
2140 IF KR=0 THEN CLOSE #9 :
: RETURN !049
2150 ON KR GOTO 2160,2170,21
80,2190,2200,2210,2160 !077
2160 IF SCH THEN PRINT #9:TA
B(10);SCH$;" Search found:"
!033
2165 IF KR=1 OR KR=7 THEN IF
 F1 THEN PRINT #9:TAB(5);C1;
" ";A1$;" ";F1$ !249
2170 IF KR=2 OR KR=7 THEN IF
 F2 THEN PRINT #9:TAB(5);C2;
" ";A2$;" ";F2$ !254
2180 IF KR=3 OR KR=7 THEN IF
 F3 THEN PRINT #9:TAB(5);C3;
" ";A3$;" ";F3$ !003
2190 IF KR=4 OR KR=7 THEN IF
 F4 THEN PRINT #9:TAB(5);C4;
" ";A4$;" ";F4$ !008
2200 IF KR=5 OR KR=7 THEN IF
 F5 THEN PRINT #9:TAB(5);C5;
" ";A5$;" ";F5$ !013
2210 IF KR=6 OR KR=7 THEN IF
 F6 THEN PRINT #9:TAB(5);C6;
" ";A6$;" ";F6$ !018
2230 IF PR OR C7 THEN P=P+1
:: PRINT #9:TAB(5);P;" ";MEM
$;" ";DT$ !191
2240 CLOSE #9 :: IF PR OR C7
 THEN PR=0 :: GOTO 1740 !039
2250 ST=1 :: GOSUB 500 :: RE
TURN !048
2260 ! ** CATALOG ** !134
2270 CALL CLEAR :: CALL SCRE
EN(8):: DIM TP$(5)!043
2280 DISPLAY AT(23,1):"what
disk to catalog?" :: ACCEPT
AT(23,23)VALIDATE("01256789A
BCD")SIZE(-1):D$ :: IF D$="0
" THEN CALL CLEAR :: ST=1 ::
 GOTO 2640 !140
2290 TP$(1)="DIS/FIX" :: TP$
(2)="DIS/VAR" :: TP$(3)="INT
/FIX" :: TP$(4)="INT/VAR" ::
 TP$(5)="PROGRAM" !195
2300 H2=0 :: FL=0 :: CALL CL
EAR !079
2310 ON ERROR 2650 :: OPEN #
12:"DSK"&D$&".",INPUT ,INTER
NAL,RELATIVE ! Read file !09
0
2320 ON ERROR 2650 :: INPUT
#12:A$,A,B,C !164
2330 FN$=A$&" AV;"&STR$(C)&"
 US:"&STR$(B-C+2):: FN2$=A$
!170
2340 PRINT A$;" AV:";C;" US:
";B-C: :!078
2350 PRINT "----------------
---------   FILENAME   SZ
TYPE       ----------------
---------" !206
2360 INPUT #12:B$,G,H,I !200
2370 HH=LEN(STR$(H)):: IF HH
<2 THEN H$=" "&STR$(H)ELSE H
$=STR$(H)!181
2380 IF G=0 THEN 2590 !035
2390 IF G<0 THEN 2420 !121
2400 P$=" " !026
2410 GOTO 2440 !224
2420 G=G*-1 !203
2430 P$=" Y" !116
2440 CC=LEN(B$):: DD$=RPT$("
 ",11):: DD=LEN(DD$):: EE=DD
-CC :: B$=B$&SEG$(DD$,1,EE)!
220
2450 IF H<10 THEN 2460 ELSE
2480 !215
2460 L$=" " !022
2470 GOTO 2490 !018
2480 L$="" !245
2490 IF I=VAL(CHR$(48))THEN
2500 ELSE 2520 !188
2500 PRINT B$;H$;" ";TP$(G)!
238
2510 GOTO 2530 !058
2520 PRINT B$;H$;" ";TP$(G);
I;!159
2530 H2=H2+H !187
2540 FL=FL+1 !157
2550 CALL KEY(3,K,S):: IF S<
>1 THEN 2580 !039
2560 IF K=82 THEN CALL CLEAR
 :: GOTO 2610 !079
2570 CALL KEY(3,K,S):: IF S<
>1 THEN 2570 !029
2580 GOTO 2360 !144
2590 PRINT :FL;" files on ds
k: ";A$ !114
2600 PRINT " total file sect
ors:";H2+2: :!044
2610 CLOSE #12 !202
2620 INPUT " another? (Y/N)
":M$ !070
2630 IF M$="Y" THEN CALL CLE
AR :: GOTO 2280 !022
2640 CALL CLEAR :: ST=1 :: G
OSUB 500 :: GOTO 1060 :: RET
URN !124
2650 CALL CLEAR :: DISPLAY A
T(21,1):"wrong drv # or dsk
not in!": :"OR CALL FILES LI
MIT P.E.T.C." :: CALL KEY(0,
K,S):: IF S<1 THEN 2650 !022
2660 ON ERROR 2270 :: CLOSE
#12 :: GOTO 2270 !242
2670 ! ** OPEN ANOTHER FILE
** !046
2680 LF=1 :: CALL CLEAR :: G
OSUB 3020 :: IF SL=1 THEN GO
SUB 3450 ELSE IF HM=0 THEN 2
710 !148
2690 IF ER THEN ER=0 :: DISP
LAY AT(20,1):ER$ !227
2700 IF (F1=0)*(F2=0)*(F3=0)
*(F4=0)*(F5=0)*(F6=0)THEN AF
```

(See Page 14)

WINDOW:1-6 —

(Continued from Page 13)

```
C,F1,F2,F3,F4,F5,F6=0 :: F1$
,F2$,F3$,F4$,F5$,F6$="" :: C
1,C2,C3,C4,C5,C6,J,J2=0 !034
2710 IF AFC THEN DISPLAY AT(
19,1):"*** all files closed
now ***" !044
2715 IF (F1>=1)*(F2>=1)*(F3>
=1)*(F4>=1)*(F5>=1)*(F6>=1)T
HEN DISPLAY AT(20,1):S22$ ::
 AFO=1 !178
2718 IF SDO OR RMD=0 THEN DI
SPLAY AT(20,1):"     three f
ile limit!" !176
2720 DISPLAY AT(21,1):"use "
"QQQ"" to escape or      ENT
ER Input File Name:" :: ACCE
PT AT(23,10):F14$ :: DISPLAY
 AT(22,1):RPT$(" ",84)!237
2730 AFO=0 :: S22$="" :: IF
F14$="QQQ" THEN F14$="" :: L
F,K=0 :: GOSUB 500 :: GOTO 1
060 !076
2740 IF F14$="QUIT" THEN GOS
UB 4500 !196
2750 IF F14$="BACK" THEN F7$
,F14$="" :: CALL CLEAR :: GO
SUB 3340 :: GOTO 50 !1342755
 CA=1 :: IF HM=0 THEN LF=0 !
135
2760 DISPLAY AT(21,1):RPT$("
 ",84):: DISPLAY AT(23,1):"W
hich Disk #?" :: ACCEPT AT(2
3,15)VALIDATE("ABCD01256789X
")SIZE(-1):D14$ !037
2770 IF D14$="0" THEN F14$="
" :: GOSUB 500 :: GOTO 1060
ELSE F14$="DSK"&D14$&"."&F14
$ !063
2780 DISPLAY AT(23,1):"To Op
en ";F14$:" o.k.? (Y/N)" ::
ACCEPT AT(24,20)VALIDATE("NY
ny")SIZE(-1):YN$ :: IF YN$<>
"Y" AND YN$<>"y" THEN 2670 !
109
2790 DISPLAY AT(24,1):"TO WI
NDOW #" :: ACCEPT AT(24,13)S
IZE(-1)VALIDATE("0123456"):W
ND !037
2795 IF WND=0 THEN F14$,D14$
="" :: CALL CLEAR :: LF,K=0
:: GOSUB 500 :: GOTO 1060 !1
32
2800 DISPLAY AT(24,1):"SINGL
E LINE OR CONT (1/0)" :: ACC
EPT AT(24,27)SIZE(-1)VALIDAT
E("013"):SL :: IF SL=0 THEN
J=1 ELSE J=0 !235
2810 LF=0 :: IF WND=1 THEN F
1$=F14$ :: F1,G1=1 :: ON ERR
OR 2880 :: OPEN #1:F1$,INPUT
 !039
2820 IF WND=2 THEN F2$=F14$
:: F2=2 :: G2=1 :: ON ERROR
2880 :: OPEN #2:F2$,INPUT !2
34
2830 IF WND=3 THEN F3$=F14$
:: F3=3 :: G3=1 :: ON ERROR
2880 :: OPEN #3:F3$,INPUT !2
41
2840 IF WND=4 THEN F4$=F14$
:: F4=4 :: G4=1 :: ON ERROR
2880 :: OPEN #4:F4$,INPUT !2
48
2850 IF WND=5 THEN F5$=F14$
:: F5=5 :: G5=1 :: ON ERROR
2880 :: OPEN #5:F5$,INPUT !2
55
2860 IF WND=6 THEN F6$=F14$
:: F6=6 :: G6=1 :: ON ERROR
2880 :: OPEN #6:F6$,INPUT !0
06
2870 HM=HM+1 :: ST=1 :: GOTO
 2890 !170
2880 ER=1 :: ER$="ERROR ENCO
UNTERED. TRY AGAIN" :: GOSUB
 4200 :: GOTO 2670 !115
2890 IF F14$="0" THEN 720 EL
SE GOSUB 500 :: GOTO 720 !18
9
2900 ! ** CLOSE A FILE ** !1
62
2910 CALL CLEAR :: GOSUB 302
0 :: GOSUB 3130 !146
2920 DISPLAY AT(22,1):"enter
 number (not name!)
                        Yo
u wish to CLOSE? " !127
2922 ACCEPT AT(24,23)VALIDAT
E("1234560")SIZE(-1):CF :: I
F CF=0 THEN 3010 !143
2925 IF (CF<>F1)*(CF<>F2)*(C
F<>F3)*(CF<>F4)*(CF<>F5)*(CF
<>F6)THEN 2922 !216
2930 IF AFC THEN CF=0 :: GOT
O 3000 ELSE ON CF GOTO 2940,
2950,2960,2970,2980,2990 !24
2
2940 IF CF=1 THEN CLOSE #1 :
: F1=0 :: F1$="" :: GOTO 300
0 !242
2950 IF CF=2 THEN CLOSE #2 :
: F2=0 :: F2$="" :: GOTO 300
0 !246
2960 IF CF=3 THEN CLOSE #3 :
: F3=0 :: F3$="" :: GOTO 300
0 !250
2970 IF CF=4 THEN CLOSE #4 :
: F4=0 :: F4$="" :: GOTO 300
0 !254
2980 IF CF=5 THEN CLOSE #5 :
: F5=0 :: F5$="" :: GOTO 300
0 !002
2990 IF CF=6 THEN CLOSE #6 :
: F6=0 :: F6$="" :: IF CA TH
EN CA=0 :: GOSUB 3340 :: GOT
O 1 ELSE GOTO 3000 !055
3000 CF=0 :: IF (F1=0)*(F2=0
)*(F3=0)*(F4=0)*(F5=0)*(F6=0
)THEN AFC=1 !248
3010 IF AFC THEN GOSUB 500 :
: RETURN ELSE ST=1 :: GOTO 5
00 :: RETURN !253
3020 ! ** SHOW FILE STATUS *
* !016
3030 IF V THEN CALL CLEAR
91
3040 DISPLAY AT(2,1):"File s
tatus: 0=closed 1=open": :TA
B(2);"1 -";F1;F1$;C1: :TAB(2
);"2 -";F2;F2$;C2: :TAB(2);"
3 -";F3;F3$;C3 !084
3045 DISPLAY AT(10,1):TAB(2)
;"4 -";F4;F4$;C4 !154
3050 DISPLAY AT(12,2):"5 -";
F5;F5$;C5: :TAB(2);"6 -";F6;
F6$;C6: :TAB(2);"7 -";F7;SF$
;"(MEMO)": :TAB(2);"last";WN
D;F14$ !156
3052 IF V THEN DISPLAY AT(22
,1):" press any key to conti
nue" !160
3055 IF RMD THEN IF V THEN D
ISPLAY AT(20,5):"sl status:"
;SL;"   ramdisk " :: GOTO 30
70 !254
3060 IF SDO THEN IF V THEN D
ISPLAY AT(20,5):"sl status:"
;SL;"   single" ELSE 3065 !2
34
3065 IF SDO=0 THEN IF V TH
 DISPLAY AT(20,5):"sl stat
:";SL;"   multiple" !0633070
 IF LF THEN 2690 ELSE IF V T
```

(See Page 15)

WINDOW:1-6 —

(Continued from Page 14)

```
HEN GOSUB 3130 !188
3080 IF AFC THEN DISPLAY AT(
20,1):S19$ !108
3090 IF V THEN CALL KEY(0,K,
S):: IF S<1 THEN 3090 ELSE I
F K=77 THEN RETURN !108
3100 IF J THEN CALL CLEAR ::
 GOTO 2720 !202
3110 IF AFC THEN GOSUB 500 E
LSE IF V THEN ST=1 :: GOTO 5
00 :: RETURN !125
3120 IF K=77 THEN RETURN ELS
E CALL CLEAR :: ST=1 :: GOSU
B 500 :: GOTO 800 !235
3130 ! ** ALL FILES O OR C *
* !138
3140 !!131
3150 IF (F1=0)*(F2=0)*(F3=0)
*(F4=0)*(F5=0)*(F6=0)THEN DI
SPLAY AT(23,2):S19$ :: F1$,F
2$,F3$,F4$,F5$,F6$="" :: SL,
J,J2=0 !097
3155 IF F1 THEN G1=1 ELSE IF
 F2 THEN G2=1 ELSE IF F3 THE
N G3=1 ELSE IF F4 THEN G4=1
 ELSE IF F5 THEN G5=1 ELSE IF
 F6 THEN G6=1 !197
3160 IF J2 THEN 650 ELSE RET
URN !014
3170 ! ** DELAY FOR SL=0 **
!045
3175 IF BB=0 THEN IF TM=0 TH
EN DISPLAY AT(22,1):"enter d
elay time" :: ACCEPT AT(22,1
8)SIZE(-4)VALIDATE("01234567
89"):TM !202
3180 IF BB THEN 3210 !023
3190 IF K=66 THEN GOTO 3220
!098
3200 FOR DLY=1 TO TM :: NEXT
 DLY !128
3210 IF PT THEN 3260 ELSE IF
 BB THEN DISPLAY AT(23,5):"s
et delay time: " :: ACCEPT A
T(23,20)VALIDATE("0123456789
")SIZE(-4):TM :: PT,J=1 :: K
=0 !087
3220 !!131
3230 !!131
3240 IF BB OR KK=82 THEN DIS
 AY AT(23,5):"reset sl=0? (
)/1)" :: ACCEPT AT(23,20)VAL
IDATE("013")SIZE(-1):SL !247
)250 IF SL=1 THEN BB,KK,J,J2
=0 :: GOTO 1060 ELSE IF K=66
 THEN 3260 !185
3260 DISPLAY AT(23,5):" mode
 continuous";TM :: IF KK=13
THEN KK=0 :: RETURN !1393263
 CALL KEY(0,K,S):: IF S<1 TH
EN 3370 :: IF SL=0 THEN KK=K
 !040
3270 IF KK=80 THEN GOSUB 211
0 :: KK=0 !031
3280 IF KK=69 THEN CA=1 :: K
K=0 :: GOSUB 3340 :: GOTO 13
50 !211
3290 HLP=1 !156
3300 IF SL=0 THEN IF J2 THEN
 RETURN ELSE GOTO 570 :: RET
URN !216
3305 CALL KEY(0,K,S):: IF S<
>1 THEN 3305 !251
3310 ! ** IF SL=0 FILS 4-6 !
215
3320 IF F4 OR F5 OR F6 THEN
IF SL=0 THEN J2,G4,G5,G6=1 :
: HLP=1 !130
3330 RETURN !136
3340 RETURN ! CLEAR IF NOT I
NITIALIZING !253
3350 F1,F2,F3,F4,F5,F6,F7,F9
,F14,J,J2,HLP,SL,SCH,C1,C2,C
3,C4,C5,C6,C7=0 :: A1$,A2$,A
3$,A4$,A5$,A6$,F1$,F2$,F3$,F
4$,F5$,F6$,D14$,F14$,D7$=""
!235
3360 F7$,MEM$,SCH$="" :: BB,
CA,ER,FR,HM,LF,PR,PT,RT,TM,S
DO,ST,WND=0 :: RETURN !153
3370 ! ** ESCAPE F1 FROM SL=
0 ** !083
3380 CALL KEY(0,O,S):: IF S<
>1 THEN 3420 !115
3390 IF O=66 THEN 3400 ELSE
3420 !072
3400 TM,J=0 :: SL=1 :: DISPL
AY AT(23,3):" mode: step-thr
ough" !037
3410 CALL KEY(0,O,S):: IF S<
>1 THEN 3410 !105
3420 IF O=80 THEN GOSUB 2110
 ELSE O=0 :: ! RETURN !149
3430 ! ** BACKUP DELAY ** !2
08
3435 IF HM=6 THEN 3445 !209
3440 FOR DLY=1 TO TM :: NEXT
 DLY !128
3445 RETURN !136
3450 ! ** CLEAR VARIABLES **
 !171
3460 IF F1=0 THEN F1$,A1$=""
 :: C1=0 !120
3470 IF F2=0 THEN F2$,A2$=""
 :: C2=0 !124
3480 IF F3=0 THEN F3$,A3$=""
 :: C3=0 !128
3490 IF F1=0 AND F2=0 AND F3
=0 THEN IF SL=0 OR SL=3 THEN
 J=0 ELSE J=1 !088
3500 IF F4=0 THEN F4$,A4$=""
 :: C4=0 !132
3510 IF F5=0 THEN F5$,A5$=""
 :: C5=0 !136
3520 IF F6=0 THEN F6$,A6$=""
 :: C6=0 !140
3530 IF F4=0 AND F5=0 AND F6
=0 THEN IF SL=0 THEN J2=0 ::
 !127
3540 RETURN !136
4000 ! ** SET MEMO FILE REF
** !002
4010 IF FV THEN DISPLAY AT(2
2,2):"display window #:" ::
ACCEPT AT(22,19)VALIDATE("12
3456")SIZE(-1):GV :: DISPLAY
 AT(22,2):RPT$(" ",56)!254
4020 IF GV=1 THEN GU$=F1$&"
"&STR$(C1)ELSE IF GV=2 THEN
GU$=F2$&" "&STR$(C2)ELSE IF
GV=3 THEN GU$=F3$&" "&STR$(C
3)ELSE IF GV=4 THEN GU$=F4$&
" "&STR$(C4)ELSE IF GV=5 THE
N GU$=F5$&" "&STR$(C5)ELSE I
F GV=6 THEN GU$=F6$&" "&STR$
(C6)!246
4030 IF GV=1 THEN GV$=A1$ EL
SE IF GV=2 THEN GV$=A2$ ELSE
 IF GV=3 THEN GV$=A3$ ELSE I
F GV=4 THEN GV$=A4$ ELSE IF
GV=5 THEN GV$=A5$ ELSE IF GV
=6 THEN GV$=A6$ !0874040 RET
URN !136
4100 ! CALL FILES WARNING !1
36
4110 IF SDO=0 THEN PRINT :"N
ON-RAMDISK MULTIPLE DRIVE  U
SERS MUST CALL FILES FOR  M
ORE THAN 3 FILES.": :!233
4120 PRINT : :"DO YOU WANT T
O CALL FILES()?": : :: INPUT
 "enter 1=YES or 0=NO ":CLF
!224
```

(See Page 16)

1995 TI FAIRS

SEPTEMBER

10th International TI-Meeting, Sept. 22-24, Wohlfahrtsgebäude der Wiener E-Werke (Welfare Building of the Vienna Electricity Board), Wachaustr. 28, A-1020 Vienna, Austria. For information write Kurt Radowisch, TI- and Geneve User Group Vienna, Grossbauerstr. 24, A-1210, Vienna, Austria.

TI New England Fall Faire, Sept. 30, Emanuel Lutheran Church, 200 Greenwood St., Worcester, Massachusetts. Contact Jim Cox, 905 Edgebrook Dr., Boylston, MA 01505 or (508) 869-2704.

OCTOBER

Chicago International TI Faire, Oct. 28, Evanston Public Library. Contact Chicago TI Users Group, P.O. Box 7009, Evanston, IL 60204-7009, or Hal Shanafield, (708) 864-8644.

1996 TI FAIRS

FEBRUARY

Fest West '96, Feb.17, Ramada Inn, 1601 Oracle Dr., Tucson, Arizona. Contact SouthWest Ninety-Niners User Group by sending e-mail to twills@primenet.com. Or call the Cactus Patch BBS at (520) 290-6277.

MAY

Multi Users Group Conference, May 25, Ohio National Guard Armory, Brookpark. Contact Glenn Bernasek, 13246 Harper Rd., Strongsville, OH 44136, or call (after 9 p.m. Eastern time)at (216) 846-0865 or Internet dd314@cleveland.freenet.edu.

This TI event listing is a permanent feature of MICROpendium. User groups and others planning events for TI/Geneve users may send information for inclusion in this standing column. Send information to MICROpendium Fairs, P.O. Box 1343, Round Rock, TX 78680.

WINDOW:1-6 —

(Continued from Page 15)

```
4130 IF CLF THEN CALL CLEAR
:: PRINT CLF$: :CLG$: :CLH$:
 :CLI$: : :CLJ$: :CLK$ :: ST
OP !190
4150 RETURN !136
4200 REM ** CATCH ERROR FILE
** !207
4210 PRINT ER$: :"FILE NOT F
OUND!": :"YOU MUST CLOSE WIN
DOW #";WND: : :: INPUT "ENTE
R WINDOW #: ":DRV !170
4220 ON DRV GOTO 4225,4230,4
235,4240,4245,4250 !243
4225 IF DRV=1 THEN F1=0 :: F
1$="" :: DRV=0 :: GOTO 4260
!082
4230 IF DRV=2 THEN F2=0 :: F
2$="" :: DRV=0 :: GOTO 4260
!085
4235 IF DRV=3 THEN F3=0 :: F
3$="" :: DRV=0 :: GOTO 4260
!088
4240 IF DRV=4 THEN F4=0 :: F
4$="" :: DRV=0 :: GOTO 4260
!091
4245 IF DRV=5 THEN F5=0 :: F
5$="" :: DRV=0 :: GOTO 4260
!094
4250 IF DRV=6 THEN F6=0 :: F
6$="" :: DRV=0 :: GOTO 4260
!097
4260 F14=0 :: D14$,F14$="" :
: ER=0 :: GOTO 2670 :: RETUR
N !219
4300 ! ** DATE OPTION ** !16
2
4310 DISPLAY AT(19,2):RPT$
 ",84):: DISPLAY AT(19,2):"E
nter Date (00/00/00)" :: ACC
EPT AT(19,14)VALIDATE("01234
567890/")SIZE(-8):DT$ :: RET
URN !013
4400 ! ** SINGLE DRIVE ** !2
31
4410 DISPLAY AT(12,1):"SINGL
E DRIVE users:  if you wish
to open an OUTPUT MEMO file,
 is there enough space" !on
your disk?" !135
4420 DISPLAY AT(15,1):"on yo
ur disk?  If unsure,  go ba
ck to program and use  ""C""
 for CATALOG to find out!" !
199
4430 SDO=0 :: RETURN !167
4500 ! END PRINT !039
4510 PRINT ND$ !082
4520 FOR DLY=1 TO 500 :: NEX
T DLY !063
4550 PRINT F1$;C1:A1$:F2$;C2
:A2$:F3$;F3:A3$:F4$;F4:A4$:F
5$;F5:A5$:F6$;F6:A6$: :"PR
RAM ENDED.":"ALL FILES CLOSE
D." :: END !197
4600 REM ** CALL/BLUE ** !02
1
4610 SUB BLUE !149
4620 CALL SCREEN(5)!150
4630 FOR L=0 TO 14 !111
4640 CALL COLOR(L,16,1)!051
4650 NEXT L !226
4660 SUBEND !168
```

Group gets new name, leader

The TI - and Geneve Users Group Vienna is being reorganized as the Danubia 99ers, as Vienna is located on the Danube River, according to Kurt Radowisch, chairman for the group.

Radowisch replaces long-term chairman Alfred Slovak, who says he will have less time in the future.

Radowisch notes that "you can count with your fingers" the group's members, not all of whom live in Vienna, but some of whom are within an hour's car trip of that city.

In September, the group was the host for Europe's 10th International TI-Meeting.

Address for the Danubia 99ers is Grossbauerstr. 24, A-1210 Vienna, Austria.

RamCharged ends 800 service

RamCharged Computer Service has discontinued its 800 toll free order number. Ron Markus of the company cited the high cost of the service as the reason for ending it. RamCharged can be reached at (216) 243-1244.

THE ART OF ASSEMBLY — PART 52

Cheap and Dirty

By BRUCE HARRISON

Without further delay, we get right into a new topic. This month, we're revisiting the subject of pseudo-random numbers. We're also reworking parts of our Video Titler, in response to a request from our friend Dick Bulmer, of Omemee, Ontario. Dick felt that our Titler could use a couple more "wipes." We started playing around with his idea, and added some of our own, ending up with 11 new wipes in the program.

One of those is not really a wipe, but a random-block replacement of one picture with the other. To do this in the limited memory available (the Titler program resides entirely in low memory) we had to invent a slightly better method of doing our random numbers. Hence the title "Cheap and Dirty" for this month's column, as the random number routine in the sidebar is not elegant, nor does it supply very high quality sequences of random numbers, but it does well enough for the intended purpose.

THE SEEDING PROCESS

To get our sequence of pseudo-random numbers started, we need a "seed" number that's unpredictable. In this case, when we enter the process, a "key loop" has been running, and that loop includes the LIMI 2 and LIMI 0 instructions. Thus the counter at >8379 will have been incrementing every 60th of a second, and neither we nor the user will be able to predict its state. Being only a one-byte counter, this has only 256 possible states, and so can yield only 256 different sequences of random numbers. That's enough, however, to give the user the impression of complete randomness in the way one picture replaces the other. We've arranged things in this case so that no block in the picture is written twice, and that the process always takes about the same amount of time, regardless of the particular seed number used. How? Well, that would be telling!

OKAY, WE'LL TELL!

We want to take each of the numbers from 0 through 191 once and only once, but in random order. We'll write the new picture into place as 192 blocks of 32 bytes each. We want to do this with minimal expenditure of time and memory. Now look at the sidebar, and we hope it will all be clear as mud in just a moment. But first, just a word or two about how things are set up in memory when the Titler is running. The program is in low memory. Assuming two pictures have been loaded as Frame 1 and Frame 2, they're in high memory from >A000 through >FFFF. Frame 1's pattern descriptions (black and white data) runs from >A000 through >B7FF. That frame's color data runs from >B800 through >CFFF.

Frame two is in two similar blocks starting at >D000 and running right to the end of memory at >FFFF. Its color part starts at >E800. When one of those is put into VDP Ram so we can see it, its pattern part starts at 0 in VDP, and its color part at >2000.

The first line in the sidebar uses a subroutine (not shown) to set R1 to point to the pattern part of whichever frame is not currently on-screen. Thus, after that subroutine, R1 will contain either >A000 or >D000, as needed. We made a kind of practical choice to take our random blocks in chunks of 32 bytes at a time. That means the whole picture is 192 such blocks, each of which occupies four character-size chunks of the screen.

Now we stash away R1 into R13, so we can change R1 and still get its original value back from R13 when we need it. We will now construct in memory a table of numbers from 0 through 191, each occupying one byte. We put this table starting at label EAUT. That's an area which gets used to store the Editor/Assembler utilities for our program, but at this point the utilities have already been moved into place in low memory, so we can use this part of memory over again, rather than setting aside a separate block. When the table is finished, the left byte of R3 contains 192, and we want that number, but in the right byte. Since the right byte is still zero, we SWPB to get R3 to contain 192.

Now we have to get a random seed, which we do by simply moving the word at >8378 into >83C0. That number will be somewhere in the range of 0 through 255. Next we use a method borrowed from TI, multiplying the number from >83C0 by 28,645, then adding 31,417 to the low-order word of that product in R5. We move that low order word back to >83C0 for next time, then discard the high order word of the product by clearing R4. At this stage, the register pair R4-R5 contains a "random" number in the range of 0 through 65,535. We don't care what it is, we're going to divide it by the number in R3. For this first pass, R3 contains 192. After division, there will be a remainder in R5 that will always be a number in the range 0 through 191. Gee, that sounds like what we were after, doesn't it? Indeed it is!

Now we take the corresponding number from our table. This first time, that will be whatever number we had in R5 to start with. Let's say it's 95, for example. We move that byte into the left byte of R0 with MOVB @EAUT(R5),R0. To get R0 set to that number as a word, we SRL R0,8. Now R0 equals 95, but we want to point to a block of bytes on a 32-byte boundary in VDP, so we multiply that number in R0 by 32 with SLA R0,5.

Still with us? It's getting tricky now. We get the base of the "Frame" from R13 into R1, then add the "offset" by simply adding R0 to R1. Now before we proceed to write this, let's review where we are. R0 points to a spot in the pattern descriptor part of VDP RAM, on a 32-byte boundary. R1 points to the corresponding place in the frame in memory that's currently not on the screen. R2 still contains 32, so when we BLWP @VMBW, four adjacent characters will get newly defined on the screen. That takes care of the "black and white" for those 32 bytes, so now we have to get the corresponding color information into place to go with that. To do that, we add >2000 to R0, add >1800 to R1, leave R2 alone, and BLWP @VMBW again.

WHAT HAVE WE DONE?

At this stage, we have the screen mostly showing one picture,

(See Page 18)

THE ART OF ASSEMBLY —

(Continued from Page 17)

but somewhere in the screen there's a four-character block from the new picture. Now hold onto your hat, because here's where it gets even trickier! We're going to remove the number we just picked from the table so that it won't be picked again. We do that in the line MOVB @EAUT-1(R3),@EAUT(R5). The number at EAUT-1(R3) is one that won't be available on the next pass, because we'll DEC R3 before going through the loop again. Thus the number that we took from the table will no longer be there. The position formerly occupied by 95 will contain 191 now. Thus if our random number process should again yield 95, it will go to the 95th place in the table, but 95 won't be in the table any more.

> The random number routine is not elegant, nor does it supply very high quality sequences of random numbers, but it does well enough for the intended purpose.

Now the table no longer contains 192 numbers, but only 191 of them (0 through 191 with 95 missing and 191 being in the place of 95), so we have to adjust for that before going for another random number. We thus DEC R3, so that it contains 191. Since R3 is not zero, we go back for another random number. This time, when we divide by R3, the remainder in R5 will be a number from 0 through 190, and our random number will be chosen from those 191 numbers still in the table.

Thus we make 192 passes through the loop that starts at label RANDN0. Each time we take a different 32-byte chunk and its corresponding color bytes from the new frame in high memory and write those into VDP, replacing that piece of the picture that was on the screen. Note that on our last pass through the process, R3 contains 1, so the DIV operation will yield no remainder, and our last number will be whatever was left at position EAUT. (That won't necessarily be zero, because zero could have been removed from the table earlier.) When R3 decrements to zero, we're finished. All 192 blocks of the new picture are on-screen, and we can branch back to a place in the program that waits for the user's next keystroke.

Just for the edification of those who are still novices in assembly, we've put into the sidebar two small Extended BASIC programs that emulate what our assembly routine does, at least as well as XB can do that. The first runs only the numbers from 0 through 10, so it takes only two seconds to run. The second one does 192 numbers, like the assembly one does, but that takes about 28 seconds.

To see how fast the assembly code does this, you'll need to get our disk "Video Titler" from our friend Dr. Charles Good (see last month's column for his address). You'll simply load in two pictures, (samples are on the disk) then view either of them, and press R on the keyboard. In less than a second, all 192 random blocks will have been selected and written to the VDP RAM.

OTHER CHANGES

The Titler disk has been updated as of Dec. 22, 1994 to include the following additional wipes:

KEY	Wipe action
J	Horizontal wipe from edges to center
C	Vertical wipe from top and bottom to middle
I	Inward spiral from outside to center
O	Outward spiral from center to edges
R	Random sequence of 192 picture pieces
Y	Venetian blind wipe downward
U	Venetian blind wipe upward
1	Corner from upper left to lower right
=	Corner from upper right to lower left
Z	Corner from lower left to upper right
	Corner from lower right to upper left

Of these, the J and C keys were as requested by Dick Bulmer, to complement the H and V keys which go from center to edges and from middle to top and bottom, respectively. When we started this attempt, we weren't sure whether the J and C key actions would fit in memory, but found those so easy that we went ahead and added the others. We still have some of low memory left, but can't think of any more wipes we'd like to try just now.

The assembly code in this month's sidebar is of course just a small part of a program, and you'd have to modify it to do your own particular job. In most cases, you'd need a block of memory set aside somewhere to contain your table of numbers. If that table is to contain more than 256 numbers, you'll have to double its size so that each number is a word instead of just a byte, and you'll have to make sure it starts on an even address. There we go again, creating yet another exercise for our serious readers. That should keep you busy till next month, when we'll try to provide another pleasant surprise. See you then.

SIDEBAR 52

```
0001  * SIDEBAR 52
0002  * CHEAP AND DIRTY RANDOM NUMBERS
0003  *   Code by Bruce Harrison
0004  *   22 Decemebr 1994
0005  *   PUBLIC DOMAIN
0006  *
0007  * PART ONE - ASSEMBLY CODE
0008  *
0009  * THIS IS JUST A 'SNIPPET'
0010  * NOT A COMPLETE PROGRAM
0011  *
0012  RNDCHG BL   @STFRM       SET FRAME ADDRESS
0013  RNDCH0 LI   R2,32        32 BYTE CHUNKS
```

(See Page 19)

THE ART OF ASSEMBLY —

(Continued from Page 18)

```
0014          MOV  R1,R13        STASH R1 IN R13
0015          CLR  R3            START WITH 0 IN R3
0016          LI   R9,EAUT       POINT AT MEMORY AREA
0017  BLDTBL MOVB R3,*R9+        MOVE LEFT BYTE R3
0018          AI   R3,>100       INCREMENT LEFT BYTE
0019          CI   R9,EAUT+192   AT END OF TABLE?
0020          JLT  BLDTBL        IF LESS, REPEAT
0021          SWPB R3            SWAP SO R3=192
0022          MOV  @>8378,@>83C0 MOV TIMER TO SEED
0023  RANDNO LI   R4,28645       LOAD A BIG NUMBER
0024          MPY  @>83C0,R4     MULTIPLY BY SEED
0025          AI   R5,31417      ADD A BIG NUMBER
0026          MOV  R5,@>83C0     MOV RESULT TO SEED
0027          CLR  R4            CLEAR HIGH WORD
0028          DIV  R3,R4         DIVIDE R4-R5 PAIR BY R3
0029          MOVB @EAUT(R5),R0 TAKE NUMBER FROM TABLE
0030          SRL  R0,8          RIGHT JUSTIFY
0031          SLA  R0,5          MULTIPLY BY 32
0032          MOV  R13,R1        GET R1 BACK
0033          A    R0,R1         ADD OFFSET FROM R0
0034          BLWP @VMBW         WRITE 32 BYTES
0035          AI   R0,>2000      POINT TO COLOR TABLE
0036          AI   R1,>1800      AND STORED COLOR
0037          BLWP @VMBW         WRITE COLOR PORTION
0038          MOVB @EAUT-1(R3),@EAUT(R5) REPLACE NUMBER
0039          DEC  R3            DECREMENT R3
040           JNE  RANDNO        IF NOT ZERO, REPEAT
041           B    @PXKEY        BRANCH TO "WAIT FOR KEY"
```

The following, in Extended BASIC, emulates the random number algorithm for numbers o-10. You can run this to see that the process really works.

```
10 RANDOMIZE
20 CALL CLEAR :: DIM A(10)
30 FOR I=0 TO 10 :: A(I)=I
40 NEXT I
50 FOR I=10 TO 0 STEP -1
60 X=RND*I
70 PRINT " "&STR$(A(X));
90 A(X)=A(I)
100 NEXT I :: PRINT
110 CALL KEY(0,K,S)
120 IF S<1 THEN 110
130 IF K<>13 THEN 30
```

This is another Extended BASIC program that emulates the above random number algorithm for numbers 0-191.

```
10 RANDOMIZE
20 CALL CLEAR :: DIM A(191)
30 FOR I=0 TO 191 :: A(I)=I
40 NEXT I
50 FOR I=191 TO 0 STEP -1
60 X=RND*I
70 PRINT " "&STR$(A(X));
80 A(X)=A(I)
90 NEXT I
```

NEWSBYTES

Harrison to release music disks

The team of Bruce Harrison and Dolores P. Werths is planning a new commercial (not public domain) release in their "Assembly Music" series. This time, according to Harrison, they are preparing Christmas music.The disks will sell for $4 each or $10 for a set of three. They are not Geneve compatible. Harrison Software is located at 5705 40th Place, Hyattsville, MD 20781.

Gaskill releases Card File 3.1

Bill Gaskill has announced the release of Card File 3.1. The program creates a computerized version of the common 3x5-inch index card system. The Card File "box" has 26 tabbed inserts, in it, lettered A to Z. Behind each insert is room to store 120 index cards, a total of 3,120 index cards. Each index card can hold 1,368 bytes of free-form text on a two-sided card, and each index card can be stored using a 34-character description.

Along with the existing ability to sort, print, edit, delete and load templates, V.3.1 offers a Quick Find program to allow searchint within a range of index cards. When an insert file is loaded and the user presses 108 to go to a page, he can now also press 1-8 from any page to go to another page. V3.1 has avoidance features at the SAVE prompt to avoid unnecessary overwrites of another file. If you press S to save, then Fctn #, only the Insert File will be written to disk. If you press Fctn X, only the Index Card will be written to disk. To provide this feature V3.1 requires you type in the word DELETE at the SAVE prompt to delete an Index Card and its Insert File reference.

Existing Card File 2.0 and above and QuickFile owners can upgrade for $5. New users can purchase the program by sending $15 to William Gaskill, 2310, Cypress Court, Grand Junction, CO 81506. Specify DSSD or SSSD disks.

MUG Conference set

The Cleveland area TI99/4A users groups (TI-CHIPs and the Northcoast 99ers) will host the 1996 Multi Users Group Conference May 25 at the Ohio National Guard Armory in Brookpark, Ohio.

The event has previously been sponsored by the Lima, Ohio, Users Group. It is free to vendors and attendees. Setup will be 3:30-8 p.m. May 24 and the conference will be 8 a.m.-5 p.m. May 25, according to Glenn Bernasek of CHIPs.

Brookpark is southwest of Cleveland, 3 minutes from IH 71, 5-10 minutes from Cleveland Hopkins International Airport and 10 minutes from Exit 10 of the Ohio Turnpike (IH 80).

For further information, or to make conference reservations, contact Bernasek at 13246 Harper Rd., Strongsville, OH 44136. Phone (216) 846-0865 (after 9 p.m. EST) or e-mail at dd314@cleveland.freenet.edu. Bernasek says all messages will be returned and urges that reservations be made as early as possible.

The 1980s Home Computer Era — Part 5

1980s saw a lot of companies try to succeed with computers

By BILL GASKILL

I have no actual count on the number of players in the Home Computer Era of the 1980s, but it would be an interesting project to try to identify them all. I remember writing a FourA/Talk article back in 1989 in which I listed 40 or 50 personal computer manufacturers alone. There's no telling how many software, peripheral and support firms have come and gone along the way. At any rate, this chapter in the 1980's Home Computer Era series touches on some of the major names in the home computer game, even if they were not actually major players. You'll notice that Texas Instruments in not among the names. The reason is the best has been saved for last. I will conclude THE 1980s Home Computer Era with the birth life and death of the 99/4A.

I have no actual count on the number of players in the Home Computer Era of the 1980s, but it would be an interesting project to try to identify them all. I remember writing an article in 1989 in which I listed 40 or 50 personal computer manufacturers alone.

COLECO — Although Coleco was a strong contender in the video game cartridge and video game machine market, they never even came close to breaking into the home computer market. They didn't get into the "swing of things" until 1983, when the low-end home computer was in its twilight. Better known for its fabulous Cabbage Patch dolls, the Connecticut-based company burned like a supernova with the fanfare surrounding the announcement of the Adam home computer, but it flamed out quickly when the computer-buying public spurned the machine like a disease.

The Adam computer was announced in June 1983, with the first units shipping in September 1983. Unfortunately, after multimillion-dollar expenditures on TV ads and full color displays in magazines, the Adam couldn't be shipped in sufficient quantities to meet the Christmas 1983 demand. On top of that, it was plagued with bugs and production defects from the outset. Never a real contender in the home computer arena, the Adam slipped into oblivion when Coleco cut its losses by abandoning it in 1985. It had 80K RAM, a digital tape drive and came bundled with a printer, word processor and Smart BASIC, all for under $600. A great description of the Adam computer can be found in the August 1983 issue of *Compute!* magazine on page 26.

MATTEL — Never a serious player in the home computer market, Mattel Toys did, however, have a couple of entries. One, the Aquarius, was announced in January 1983 at a retail price of just under $200. It came with 4K RAM expandable to 52K in 4K and 16K increments with plug-in cartridges. Microsoft BASIC was built in, which meant it did not have sprites, nor did it have full-screen editing. Aquarius had a single voice sound chip, was powered by a Zilog Z-80A CPU and the CP/M operating system. The 49-key keyboard was constructed in a manner somewhere in between the membrane keyboard of the Atari 400 and a full-travel keyboard. It supported a 40-column by 24-row display.

A Mini-Expander was planned for the Aquarius that added three-voice sound, two cartridge slots (one for memory expansion, the other for software), two joystick controllers on eight-foot cords, and hookups for a cassette recorder and printer. I have no information to confirm or deny the production of the expansion system.

PCjr — IBM's "Peanut" as it was code-named, was announced in late 1983, but didn't actually become available until early 1984. It was an Intel 8088-based machine that had 64K RAM expandable to 128K (later to 512K with the addition of a new power supply and optional IBM enhancements), two cartridge slots, 40-column RGB video output and a chiclet keyboard that everyone seemed to hate. The keyboard was replaced in late summer '84 and the memory enhancements were also made available, but the machine never caught on because it only ran at 4.77mh the way it used memory made it incompatible with software written for the IBM PC/XT/AT lines and for other reasons. It faded into oblivion in a scant two years after its release, never to be heard from again.

SEGA — Perhaps flushed by the success it enjoyed in the video game market, in October 1990 Sega announced that it would release a home computer by the spring of 1991 that was compatible with both the IBM PC/AT and the Sega MegaDrive game computer. The announcement was made following an agreement signed between Sega and IBM Japan that would allow Sega to receive IBM PC/AT motherboards. The new home computer was to have two 16-bit microprocessors, an 80286 and a 68000, as well as 512K RAM expandable to 2.5 megabytes of RAM. I've never seen the computer in the U.S., but I suppose it could have been produced for sale outside the U.S. only.

SPECTRA VIDEO — Here is another minor player in the home computer market of the early '80s. Spectra Video was a

(See Page 21)

HOME COMPUTER ERA —

(Continued from Page 20)

New York firm with a factory in Hong Kong that produced its Spectravision video game machine. It was competition for the Atari VCS, Mattel's Intellivision, Coleco's Colecovision and the like. In February 1983 Spectra Video announced the impending release of its first home computer, the SV-318. The SV-318 was a non-contender in the scheme of home computer sales, but Texas Instruments did list it as a machine that the 99/4A outsold, when doing dealer sales promotional material. I have no idea if the SV-318 was ever actually produced because I've not actually seen one in the flesh or on a retailer's shelf. An item in the July 1984 issue of *Byte* magazine on page 10 tells us that Bondwell Industrial Co. Inc. planned to buy Spectra Video and then add a $995 Z80-based portable to the computer line that sported 128K and two single-sided floppies. Maybe this change in ownership changed the fate of the SV-318. Anybody out there ever actually see or own an SV-318?

The SV-318 retailed for $299.95, came with 32K RAM, half of which was VDP RAM like the TI-99, because it used the same TMS9918A chip, bought from TI, for video controlling, but unlike the TI-99, the VDP RAM in the SV-318 was accessible with PEEK and POKE commands built into the SV-318's extended Microsoft BASIC. The computer came as a slim-line white plastic console with 71 calculator-style keys, a built-in joystick in the lower right corner of the console and a cartridge port on the top near the back of the unit. It had 10 programmable function keys and 52 graphic symbols similar to the VIC-20. As 99ers would suspect, the SV-318 display sported a 32-column by 24-row graphics mode and a 40-column by 24-row text mode. It used a Zilog Z-80A chip for the CPU with CP/M for the operating system. Basic storage was via cassette, but accessories were planned that would allow disk drives, 80-column displays and other options to be plugged into the unit.

TANDY — Although Tandy was among the first to enter the personal computer market, it had almost no impact in the home computer arena. The Color Computer, or CoCo, as it was called (yech!), was discontinued in 1993, which may have made it the oldest computer from the 1980s Home Computer Era in terms of survival. In my opinion, that's about all you can say for it. Please, no letter bombs from those of you who own one.

I only recently discovered that Tandy is out of the computer business, meaning they no longer make a single computer! What a shocker it was for me to learn that the company who was among the first to give birth to the personal computer no longer makes them.

TIMEX/SINCLAIR — I never really considered the Timex/Sinclair and Sinclair computers players in the home computer market, and perhaps because of this know little about them.

• **Sinclair QL** — Announced in January 1984, a scant week before the Apple Macintosh, this Motorola 68008-based computer didn't actually appear until April, and even then it was an incomplete product with its QDOS being shipped in an interim version because the final product was not complete. The QL, which stood for Quantum Leap, sported 128K RAM and four built-in productivity programs (the Quill word processor, Abacus spreadsheet, Archive database and Easel business graphics), each with a common pull-down menu interface to make learning them easier.

The computer had two microdrives (kind of like the hex-bus wafer tapes that TI was going to produce for the 99/2 and did produce for the CC40), dished keys that were dead and unresponsive to the touch and a bus extension slot for cartridge use (but the operating system used it in the review I read), plus SuperBasic, a "very, very slow" version of the BASIC programming language.

Because of their price, the Clive Sinclair-produced computers seemed to have created a market of their own. Most of Sinclair's efforts sold well outside the U.S., but they just couldn't seem to steal the media attention that the American-manufactured machines could, and thus never spent as much time in the limelight as the Commodores and TIs. This is not to say that T/S computers didn't sell here in America, they did. According to what I've read, the ZX-80 and ZX-81 sold more than 300,000 units by the end of 1982, but I don't know when they first appeared. Besides the ZX 80 and 81, Sinclair also introduced the T/S 1000; the T/S 2000, called the ZX Spectrum elsewhere in the world; the T/S 2068, a $199.95, 72K machine released in March 1984; and the Quantum Leap computer mentioned above.

VIDEO TECHNOLOGY — I don't think this company, which had two computer factories in Hong Kong, ever actually produced the VZ200 they said they were going to sell in the United States. At least I never saw one, nor have I read anything about the VZ200 anywhere after the company's January 1983 announcement at the winter Consumer Electronics Show.

The VZ200 was to be the first home computer Video Technology produced, and it was to have sported 4K RAM expandable in 16K increments up to 64K, 12K of ROM with built-in Microsoft BASIC, one-touch entry of BASIC command a la the Timex Sinclair, a Z80 CPU, a real moving key keyboard, a 32-column by 16-row text mode display with nine colors. Retail price was set at $99 to be competitive with the Timex Sinclair, which was the market the VZ200 was trying to break into.

Video Technology also announced in January 1983 that it would begin marketing its CreatiVision game-playing computer in the United States, but I don't think that ever happened either. The CreatiVision had been marketed in Europe and Australia for over a year before the plan to invade the U.S. market was announced. However, I suspect the intense competition in the U.S. scared Video Technology off and it wisely chose to stay in a market with proven success. CreatiVision was to have retailed for $189.

THE PC INVASION — The home and personal computer market of 1995 didn't develop overnight. It has been in the works for at least 10 years, but some couldn't or wouldn't see the handwriting on the wall. Like it or not, you have only three choices of computers today, at least

(See Page 22)

HOME COMPUTER ERA —

(Continued from Page 21)

by CPU type. Two of the choices come from Apple, one in the form or their Motorola 68000 series Macs and the other in the form of the new PowerPC based Macs. The third choice is a PC running in a Windows-based DOS environment. Regardless of the manufacturer, or the CPU (80286, 80386, 80486, Pentium, Nx586), the basic computer is the same, because the power plant under the hood is the same Intel, or Intel-like chip. There are no more Atari, Commodore, Coleco, Mattel, Texas Instruments or even Tandy home computers to choose from because they have all been replaced by PCs or Macs.

"...A lot of changes have been going on in the PC market lately, some of which are bringing IBM and its compatibles much closer to home. The first change is price. It's come to a point where you can buy a generic PC system with two 5 ¼-inch floppy drives and a monitor for $600-700 and prices may go even lower for Christmas. At these prices, such machines are cheaper than some traditional home computers such as the Apple IIc. The inroads the PC compatibles have made into the home market is reflected in the recent increase in non-business software for the PC. As prices for PC compatibles spiral downward, there has been some speculation that IBM itself will soon make a serious entry into the home market, or drop out of lower-end retail sales entirely and concentrate on the high-end AT line.

"Even if the price is right, first time users may not find the PC clones user-friendly enough. But help is on the way. Microsoft, the producer of the PC operating system, seems determined to 'Macintize' the PC with its Windows software, which provides a mouse-driven user interface with pull-down menus and icons. Microsoft has been lobbying strongly with the makers of graphics coprocessor chips, display adapter cards and clones to include Windows as an integral part of the hardware design of future MS-DOS machines, and it is said to have even included Windows' graphics kernal as part of version 5.0 of MS-DOS. Putting Windows into hardware would give it the power to run efficiently even on very inexpensive computers and would help to make the systems accessible to a much wider audience." (Sheldon Leemon, *Compute!*, November 1986, p.66).

"As noted earlier, the MS-DOS invasion was very much in evidence at CES. Although Atari's PC compatible got the jump on several rivals because of Atari's surprise introduction and the low pricing, Commodore and other computer manufacturers showed PC clones that they are pushing aggressively in the U.S. market. Several companies, including Tandy, Blue Chip Electronics and Leading Edge, have gotten a big head start in the PC-clone distribution and visibility races. Commodore, Atari and the other entries in this fight will have to work hard to establish themselves." (Selby Bateman and Tom R. Halfhill, *Compute!*, April 1987, p.25).

The two excerpts above are just a small part of the handwriting on the wall that warned of the PC clone invasion. Unfortunately, neither Atari nor Commodore was ever able to overcome the force of the invasion and are victims of it today. Atari is still in the game machine business, though barely by all accounts, but it is out of the computer business, and Commodore, as we learned in Part 3 of this series, went out of business in early 1994. Apple is still very much in business today, but it also never became embroiled in the low end of the home computer market, so many of its resources were not spent fighting the war that Atari and Commodore were forced to wage against PCs after having also fought each other. But as we saw in Part 1 of this series, Apple too is now fighting what seems to be an uphill battle against a computer standard (a Windows-based PC) that outsells it six to one every year.

As we now know, the head start that the PC clone invasion had over Apple, Atari and Commodore would never be overcome. Though none of us knew it in 1987, that Consumer Electronics Show was the beginning of the end for the 1980s Home Computer Era.

Further evidence of the domination of the PC in today's home computer market can be seen by comparing the Christmas 1983 J.C. Penney catalog with its 1993 counterpart. In 1983, the catalog featured the Atari 600XL, the 800XL, the TI99/4A, the Atari 5200, the Coleco Adam, the ColecoVision machine, the Gemini VGS, Mattel's Intellivision II and the Vectrex video game system. By 1993, the picture had changed so drastically in the market for computers sold to home users that a Packard-Bell PC clone was the only entrant in the entire catalog.

As you may have figured out by now, if Texas Instruments had decided to produce the Computer 99/8, it would have been a dinosaur within a year and probably would just have become another TI orphan. Instead, TI chose to produce the TI Pro, which was a great step in the right direction, just not taken well enough. Regardless, when looking back at the remains of the 1980s Home Computer Era it is pretty obvious that, of all the players, Texas Instruments may have had the clearest crystal ball. Its decision to leave the low-end market totally was absolutely the best thing it could have done, at least from a financial point of view.

Former NET 99ers president dies

William Birdsong (Bill) Duncan, a former president of the NET (North East Texas) 99ers and a member of the Dallas TI Users Group, died Sept. 3 in Bedford, Texas. He was 83.

He was born July 2, 1912, in Dallas. While in his early 20s, he helped build the electronics of Radio Station KDNT in Denton, Texas. He worked for the Federal Aviation Agency. During World War II, this work took him into combat situations as a civilian while he was setting up air traffic control facilities in South Pacific air bases.

PC99 vs. the Red Baron

Breaking down the defenses of a worthy adversary

By MIKE WRIGHT

Manfred Freiherr von Richthofen was born May 2, 1892. In 1916, during World War I, he became a German fighter pilot. As the war progressed, he grew to fame as the Red Baron, so-called because of his red Fokker triplane, and was credited with shooting down 80 Allied planes in dogfights. He died on April 21, 1918, when he himself was shot down.

In the TI world, the program SPAD XIII from Not Polyoptics puts you at the helm of a SPAD XIII biplane and lets you do battle with the legendary Red Baron. (SPAD is an acronym for: Société Pour Aviation et ses Dérives, a French company run by aviator Louis Bériot, that built the plane.)

SPAD XIII was released in January 1987 according to MICROpendium (3:10:6). It was supplied on disk, required 32K memory expansion, and autoloaded under Extended BASIC. However, it was extremely difficult to make backups of the disk since the program was heavily protected.

We at CaDD Electronics were faced with the problem of dealing with this protection when one of our PC99 customers sent in some TI disks to be converted for use with PC99. Using the conversion utility supplied with PC99, we had absolutely no luck with the SPAD disk. Eventually this grew to be a challenge — a leather flying gauntlet slap in the face, so to speak. After all, as far as we know, there are no programs that run on a standard 4A that do not run under PC99. We wanted to preserve our 100 percent record.

The first thing we did was to copy the SPAD disk to a PC using the RSECTOR and WSECTOR utilities supplied with PC99. These utilities copy disks by reading and writing sectors. RSECTOR soon reported an error reading sector 2, and then got an error with every sector from 135 through 359.

We then used Disk+Aid to sector read the original disk. We examined sector 0 and found at offset >10 the bytes >50 and >28. The >50 ("P" in ASCII) means the disk is protected from Disk Manager copies. The >28 means there are 40 (decimal) tracks on the disk. We then used Disk+Aid to confirm what RSECTOR had shown us. Essentially, only the first 15 tracks (15 x 9 = 135 sectors) were formatted. Of these, sector 2 could not be read.

This meant that the byte representing the number of tracks had been patched to fool copy programs into believing that this was a "normal" disk.

If you use the original SPAD disk on a 4A and try to do a Disk Manager catalog, it will report that there are no files on the disk. Similarly, in PC99 you can use the DSKDIR.EXE utility, which allows you catalog TI "disks" from DOS. This, too, follows Disk Manager practice and reports no files on the disk.

The trick used here is to put two null bytes at the start of sector 1 — the directory link sector. This contains a series of two-byte entries, pointing to sectors which have File Descriptor Records (FDRs). The Disk Manager catalog routine reads sequentially from the beginning of sector 1 and exits when two null bytes are found. If there are two nulls at the beginning of the sector, then it believes there are no files on the disk. For normal usage, DSKDIR.EXE emulates this behavior. However, DSKDIR.EXE has an override switch for situations like this. If you do:

>DSKDIR DSK1 -d -x 4

You will tell DSKDIR.EXE to try to find four FDRs on the disk, even if some of the entries are null. You have to be careful doing this, because you can expect some garbage. In the case of the SPAD disk, DSKDIR.EXE returned the following:

No.	FDR	Filename	Size	Type	P
001	>000	SPADXIII	21324	PROGRAM	Y
002	>004	LOAD	34	PROGRAM	Y >006 033
003	>005	LOAD24K	97	INT/VAR254	Y >027 096
004	>000	SPADXIII	21324	PROGRAM	Y

Note that entries 1 and 4 are bogus, but that there are two real files on the disk: LOAD, and LOAD24K. DSKDIR.EXE also shows that LOAD starts in sector >006 and is 33 sectors long, while LOAD24K starts in sector >027 and is 96 sectors long.

When you tell Extended BASIC to RUN a program, it uses a different mechanism from the Disk Manager to find the filename. Instead of starting at the front of the directory link sector and searching sequentially, it starts at the middle. If nothing is found, it goes halfway back and so on until it finds an entry. It then looks up this entry and, if the name found is greater than the name it is looking for, it will go halfway back again. Similarly, if the name found is less than the name it is looking for, it will go halfway forward. This binary search is considered to be faster than searching sequentially through the FDRs.

So if you put the SPAD disk in DSK1 and start Extended BASIC, then XB's autoload mechanism will find DSK1.LOAD and execute it — even though there are apparently no files on the disk.

This also means that you can start Extended BASIC, insert the SPAD disk in DSK1, and do OLD DSK1.LOAD. If you then try to do a LIST, you will get a PROTECTION VIOLATION. This can usually be overcome by:

```
CALL INIT
CALL LOAD (-31931, 0)
LIST
```

But, if you do this, Extended BASIC crashes. The reason is that the SPAD LOAD program has had each BASIC line length byte removed and replaced with >00. It is surprising, but Extended BASIC does not need the line lengths to run. However, one or more lines without a length byte just blows LIST away.

At this point we used the DSKOUT.EXE utility on the PC. This allows you to extract a BASIC file from the TI "disk" and create a DOS file. Then we ran the BAS2ASC.EXE (BASIC to ASCII) utility on the extracted file. BAS2ASC.EXE knows about BASIC lines without a length byte, and will re-create the value. So now, we had a listing of DSK1.LOAD.

```
90 DISPLAY AT(10,7) ERASE ALL:"SPAD XIII Mk.
2" 100 CALL INIT
110 CALL LOAD(8196,255,208)
120 CALL LINK("QS")
130 DISPLAY AT(22,11):"LOADING"
```

(See Page 24)

SPAD —

(Continued from Page 23)

```
140 RUN "DSK.SPADXIII.LOAD24K"
```

In addition to these BASIC statements, the file LOAD also contains a large amount of embedded assembly code. In fact, the total file size is about 8K. So when this program is loaded, the line number table and a whole lot of assembly code are stored starting at the end of memory (>FFFF), while the actual BASIC code is stored starting at >A000.

The CALL INIT tests to see that the Memory Expansion is available, and loads TI utilities into low memory (>2000 - >3FFF). These include VSBW, VMBW, VSBR, VMBR, and VWTR. (These are described on page 248 of the Editor/Assembler manual.)

The CALL LOAD is a clever trick. This translates to storing at address >2004 the value >FFD0. Address >2004 is used by the BASIC interpreter to store the last free address in low memory, and is used when searching the REF/DEF table. Usually, this value will never exceed >3FFF. Remember that the 32K memory peripheral consists of two disconnected segments: low memory (>2000 - >3FFF) and high memory (>A000 - >FFFF). This CALL LOAD tells the interpreter that the end of low memory is at >FFD0, which is actually pretty near the end of high memory.

At this stage we ran PC99. At the title screen we entered the mini-screen debugger and set a watchpoint: >2004 = >FF. This means a PC99 break will occur any time the value >FF is written to >2004. We then pressed a key, selected Extended Basic and waited for the break to occur.

We then used the one of the PC99 memory windows to examine CPU RAM. We did a search for QS and found that it existed in three places. However, at >FFD0, there was a REF entry which showed that the executable address of QS was >FEBE. This REF was created by the embedded assembly code in LOAD.

We then set a break at >FEBE and when this occurred we went into step mode. This allowed us to write down each instruction that was being executed. Listing 1 shows the disassembled code.

LISTING 1

```
FEBE 020c LI R12, >1000cru base address
FECO 1000
FEC2 1e00 SBZ 0 turn off
FEC4 022c AI R12, >0100point to disk controller
FEC6 0100
FEC8 1d00 SBO 0 turn it on
FECA c060 MOV @>400a, R1        DSRLNK low level = >4010
FECC 400a
FECE 8821 C @>0004(R1), @>feaa >feaa = 0110 = r/w sector
FED0 0004
FED2 feaa
FED4 16f6 JNE >fec2     if r/w sector not there, try
again
FED6 c821 MOV @>0002(R1), @>feaa       = >5b38 = routine
>10 = r/w sector
FED8 0002
FEDA feaa
FEDC 0200 LI R0, >040A
FEDE 040a
FEE0 c800 MOV R0, @>8356        VDP addr for dummy PA
>0110
FEE2 8356
FEE4 0200 LI R0, >01ff >01 = drive, >ff = ?? >01 = read
FEE6 01ff
FEE8 c800 MOV R0, @>834c        drive in left byte, flag
in right
FEEA 834c
FEEC 0200 LI R0, >1000
FEEE 1000
FEF0 c800 MOV R0, @>834e        VDP addr of input buffer
FEF2 834e
FEF4 04e0 CLR @>8350    sector number 0
FEF6 8350
FEF8 c260 MOV @>feaa, R9        saved addr = >5b38 = r/w
sec
FEFA feaa
FEFC 0699 BL *R9        read sector 0 (SPADXIII...)
FEFE 1000 NOP   skip over bumped return addr
FF00 0200 LI R0, >1010 VDP address
FF02 1010
FF04 02a1 STWP R1       CPU start address (>83e0 = R0)
FF06 0202 LI R2, >0002 number of bytes to read
FF08 0002
FF0A 0420 BLWP @>202c   (VMBR vector = 2038/24aa)
FF0C 202c
FF0E c2e0 MOV @>0002, R11       [this seems to do noth-
ing]
FF10 0002
FF12 0280 CI R0, >5028 50 = "P", >28 = 40 tks/side
FF14 5028
FF16 1646 JNE >ffa4     error exit
FF18 0200 LI R0, >0003 sector number (3 = copyright)
FF1A 0003
FF1C c800 MOV R0, @>8350        sector number 3
FF1E 8350
FF20 C260 MOV @>feaa, R9        = >5b38 = r/w sec
FF22 feaa
FF24 0699 BL *R9        read sector 3
FF26 1000 NOP
FF28 0200 LI R0, >1028 VDP address (>20, >e5, copyright
end
FF2A 1028
FF2C 0201 LI R1, >feac CPU start address
FF2E feac
FF30 0202 LI R2, >0002 number of bytes to read
FF32 0002
FF34 0420 BLWP @>202c  VMBR. >feac = >20, >e5
FF36 202c
FF38 6060 MOV @>febc, R1        = >feae
FF3A febc
FF3C c2eo MOV @>0002, R11       [this seems to do noth-
ing]
FF3E 0002
FF40 c831 MOV *R1+, @>8350      sector number. R1=>feb0.
>8350=>0001
FF42 8350
FF44 132f JEQ >ffa4     fails if end of table = >0000
FF46 c801 MOV R1, @>febc        = >feb0
```

(See Page 25)

SPAD —

(Continued from Page 24)

```
FF48 febc
FF4A c260 MOV @>feaa, R9        = >5b38 r/w sec
FF4C feaa
FF4E 0699 BL *R9        read    sector    (order    =
1,4,5,6,7,8)
FF50 1000 NOP
FF52 0200 LI R0, >0003 sector number (3 = copyright)
FF54 0003
FF56 c800 MOV R0, @>8350        sector number
FF58 8350
FF5A c260 MOV @>feaa, R9        = >5b38 = r/w sector
FF5C feaa
FF5E 0699 BL *R9        read sector
FF60 1000 NOP
FF62 0200 LI R0, >1028 VDP address
FF64 1028
FF66 02a1 STWP R1       >83e0 = cpu address
FF68 0202 LI R2, >0002 number of bytes to read
FF6A 0002
FF6C 0203 LI R3, >FF88
FF6E ff88
FF70 0420 BLWP @>202c  VMBR
FF72 202c
FF74 2820 XOR @>feac, R0        >feac = >20e5; "real" sec
3 = >20e5
FF76 feac       "fake" sec 3 = >e5e5
FF78 13df JEQ >ff38     if "real" sector found then loop
FF7A 64c0 S R0, *R3     R0 = >c500, *R3 = >cc70
FF7C a8c0 A R0, @>0008(R3)
FF7E 0008
FF80 a8c0 A R0, @>000e(R3)
FF82 000e
FF84 1e00 SBZ 0
FF86 0200 LI, R0, >c397
FF88 c397
FF8A a020 A @>8384, R0
FF8C 8384
FF8E 0201 LI R1, >5b00
FF90 5b00
FF92 c081 MOV R1, R2
FF94 c110 MOV *R0, R4
FF96 41c4 SZC R4, R7
FF98 cc44 MOV R4, *R1+
FF9A 0640 DEC R0
FF9C 0642 DEC R2
FF9E 16fa JNE >ff94
FFA0 0460 B @>006a      return to GPL interpreter
FFA2 006a
FFA4 0200 LI R0, >a000 error exit
FFA6 a000
FFA8 0f40 CLR *R0+      clear all of memory
FFAA 0280 CI R0, >ffaa from >a000 - >ffaa
FFAC ffaa
FFAE 16fc JNE >ffa8
FFB0 0420 BLWP @>0000   branch to title screen
FFB2 0000
```

The first thing the assembly code does is to probe for the disk peripheral (>FEBE - >FED6). When found, the code uses the hard-coded address >400A in the peripheral ROM to find the address of the card's read-write sector routine. This is saved at address >FEAA.

The code then sets up to do a read of sector 0 into VDP memory at >1000. There is a minor irregularity here which uses >01ff instead of >0101 as the drive number and flag (read), but we think this is an attempt to obfuscate the code. The BL *R9 at >FEFC then reads the sector. (If you do not understand how to do direct sector reads, there was a short article by Dick Vandenberg in *Computer Shopper*, 84:11:152, that explained the procedure.)

The TI VMBR loaded by CALL INIT at vector >202C is then called. It loads 2 bytes from VDP address >1010 into >83E0, which is the calling workspace's R0. The value read is then compared to >5028. The program is determining if the disk is protected (>50 = "P"), and if it contains 40 (>28) tracks. If not, the code jumps to an exit routine at >FFE4. Since the real disk only has the first 15 tracks formatted, this is another protection device.

Next, sector 3 is loaded. This contains a copyright notice and the code depends on the fact that this is of a specific length. It probes at offset >28 and reads the two bytes into address >FEAC. These two bytes are >20 (a space) and >E5 (the value used by TI to fill a blank formatted sector).

A loop now starts at >FF38. Address >FEBC is a changing value that originally contains >FEAE. The memory starting at >FEAE contains a table:
>00 >01, >00 >04,
>00 >05..., >00 >00.

This table is a series of sector numbers: 1, 4, 5, 6, 7, 8, and ends with two null bytes. The code now reads the table value. If it is 0, it jumps to the error exit. If nonzero, it uses that value to read a sector. The contents of the sector are then simply ignored.

The code at address >FF52 then sets up to read sector 3 (the copyright sector). The code at address >FF62 then reads two bytes from offset >28 of the copyright sector and compares them with the previously saved value of >20E5. If they match, the code loops back to >FF38.

For quite some time we were totally baffled. The code always failed, since it always reached the end of the sector table starting at >FEAE. We had expected there to be a check for the missing sector 2, but there was none.

We now went back to the 4A and used Miller's Graphics Advanced Diagnostics to read in track 0 of the SPAD disk. We connected a PC to the 4A on a serial line, set up the PC with ProComm Plus (a terminal emulator), and set Diags to output screens to the TI RS232 port. We then dumped each of Diags' buffers to the PC. We did this twice, once in hex format and once in ASCII.

We then checked the inter-sector information. This contains, among other information, an address mark (>FE), followed by four bytes: the track number, side number, sector number, and AU size (>01 = 256 bytes/sector). We used a PC editor called Brief to search for all >FEs in the file and found, sure enough, that there was no sector 2. Instead, there was a second sector 3! We will distinguish these sector 3s by calling them "real" and "fake." The"fake" sector 3 is where sector 2 should have been, according

(See Page 26)

SPAD —

(Continued from Page 25)

to the interlace. We now constructed the following table:
Sector contents in interlace order:

0 SPADXIII... DSK
7 >01 >0a
5 LOAD24K
3 Copyright (real)
1 >00 >00 >00 >04 >00 >05
8 >04 >E2
6 >FB >97
4 LOAD
3 (2) Copyright (fake)

The contents of the fake sector 3 looked the same as the real sector 3. So we returned to PC99, put SPAD in DSK2, and loaded Disk+Aid from DSK1. Note that there is no magic about Disk+Aid. We just happen to use this as a sector editor since we are familiar with it. We also purchased the source code for this program many years ago allowing us to make changes to the program if we need to.

We then copied real sector 3 to real sector 2 on our "disk." We then used a dump program to find the sector address marks in the PC99 SPAD .DSK file. We then used the PC99 utility PATCH.EXE to change this value from >02 to >03.

We now felt very confident that the program would run. But, once again, the Red Baron put us into a tailspin, as the code returned to the title screen.

By now, we were a little desperate. However, we had a niggling feeling that the program must depend on something in the fake sector 3. We again used Brief to look at the dumped file, but this time we used two windows, so that we could see the two sectors simultaneously. Then we spotted it! There was a one-byte difference in the two sectors. The real sector 3 had an extra space making offset >28 to be >20E5, while the fake sector 3 had >E5E5 at offset >28.

Then it dawned on us. The code depends on the hardware and the interlace order. It reads a sector (from the table) and then always reads sector 3. The disk controller eventually reads the fake sector 3 instead of the real one.

The code at >FF7A deserves some mention. While looking through hex dumps of the SPAD disk we didn't see much obvious assembly code. After a while you expect to see things like >0200 (for LI, R0), etc. We finally figured out that large portions of the SPAD code are stored in reverse byte order. Again, this is done as a protection device to prevent disassembly. The code at >FF7A reads the reversed bytes, and puts them into proper order in memory. The count of the bytes depends on the XORed value from the fake sector 3. Very tricky!

The error routine at >FFA4 is also rather nasty. It erases all of memory. This prevents you from using CALL LOAD to examine memory after the program fails.

Armed with our new observations, we once again patched the SPAD .DSK file and changed fake sector 3 to match the original. Now we were cookin'. With guns primed we once again challenged the Red Baron — and once again we were shot down.

We now got to thinking about the way PC99 emulates the disk controller. We realized that, as the emulation is currently structured, we would never get the SPAD code to execute correctly. PC99 reads in a "track" of information, and then searches sequentially for the first matching sector address. So even though there are two sector 3s on the track, the emulation would never find the second one, since we do not emulate seeks that depend on interlace order.

At this point we were virtually out of ammunition. We decided against rewriting the disk emulation code to handle this one special case. Instead, we decided to patch the SPAD code to defeat the protection. The read from fake sector 3 will return >E5E5 in R0. At address >FF74 this value is XORed with >20E5. The result would be >C500. Since all of the reads are simply designed to wait until the fake sector is read, we figured we could replace the XOR with:

FF74 0200 LI R0, >C500
FF76 C500

This was surely our last chance! We started PC99, set a break at >FEBE and allowed SPAD to load. We then went to a memory window and replaced the XOR with the LI. We then hit continue. Since the debugger is inherently slower than the accelerated version of PC99, there were many anxious moments. The first hint of success was that we saw the message "LOADING" appear at the bottom of the screen. This meant we had exited the LOAD program, and were now running LOAD24K. A few seconds later the SPAD title screen appeared.

We pressed the joystick fire button and saw the guns of o... SPAD fire. In our mind, the Red Baron's Fokker triplane plunged to the earth and self-destructed. Our worthy adversary had finally been defeated. To put the seal on the event, we again dumped the SPAD disk and found the bytes that corresponded to the memory patch. We then sector edited them so that the LI permanently replaced the XOR. It was now possible to run SPAD normally under PC99. We had completed our rout of the Red Baron, and maintained our 100 percent compatibility record.

SUMMARY OF PROTECTION MECHANISMS

1. Only the first 15 tracks of the disk were formatted. The byte containing the number of tracks was then patched to be 40 (>28).
2. The disk was protected against copy by the Disk Manager.
3. Track 0 had no sector 2. Physical sector 2 was replaced with "fake" sector 3. Fake sector 3 had a one-byte difference from real sector 3.
4. The directory link sector started with two null bytes. This fools the Disk Manager into thinking there are no files on the disk.
5. The LOAD program was protected using Extended BASIC protection.
6. The LOAD program had all the line length bytes removed, which prevents it from being listed even after the XB protection is removed.
7. The last free address in low memory is patched to point to the end of high memory, which ties to code embedded in the LOAD program.
8. The CALL LINK calls a small assembly language program that was embedded in the LOAD program. Any errors detected by

(See Page 27)

SPAD —

(Continued from Page 26)

assembly code cause all of memory to be wiped out.

9. The assembly code checks to see that sector 3 has the SPAD copyright and that it is the correct length. If not, it fails.

10. The assembly code checks to see that sector 0 has the Disk Manager protection and 40 tracks. If not, it fails.

11. Most of the loaded assembly code is in byte-swap format making it hard to disassemble.

12. The main protection loop depends on the fact that the fake sector 3 will eventually be read by the hardware.

Comments on protection:

We believe the programmer for SPAD was Larry Hughes, formerly of QS Software. We would like to tip our flying helmet at his ingenuity in devising the above scheme. It certainly was a challenge to unravel it.

However, we believe that software developers have a responsibility to users of their product. This responsibility includes the ability to use the software in a convenient and nonintrusive manner. The user should also be permitted to make backups of the software to guard against media failure.

As an example, Millers Graphics Advanced Diagnostics was probably the most sophisticated disk exploration tool available for the 99/4A. Yet the disk is heavily protected and most unsophisticated users would not be able to copy it. Some years ago MG left the 4A world. If you now find that Diags will no longer load you will probably not be able to get a replacement copy and your investment in this product will have been lost.

It is our contention that software should not be protected, and should be as open as possible. We follow this policy with PC99, even though we know that some illegal copies exist. We prefer to stick to our principles and take the monetary loss rather than inconvenience our users. Besides, nearly all software protection schemes can be broken, so there is little point in using them.

It was not our intention to show you how to "break" SPAD. Even with the above information it would still be quite difficult to do this on a 4A. There are also some "holes" in our knowledge of how certain things were done. For example we do not know how the fake sector was created, or what machinations took place to pack and byte-swap the assembly code. [If anyone knows Larry Hughes, it would be interesting to show him this article and see if, after all this time, he would be willing to reveal this information.]

Instead, we were trying to satisfy a customer's needs — someone who owned a legal copy of SPAD and wanted to see it run under PC99. In doing this, we were able to illustrate the power of the PC99 debugger, and the utilities that are supplied with PC99. Armed with these, a software developer has the most formidable array of tools yet assembled to help in developing sophisticated applications for the 99/4A and allow examination of the inner workings of the machine.

We also do not want to leave you with the impression that the procedures involved were quick and easy. The whole process was spread over three weeks and took place at home, over lunch at the office, and during a business trip to Singapore (courtesy of our laptop). However, we would rather have devoted this time to enhancing PC99.

If there were any positives, apart from now being able to run SPAD, it did cause us to make minor improvements to: BAS2ASC.EXE (and the related IV2ASC.EXE, for internal/variable 254 files); DSKDUMP.EXE and DSKCHECK.EXE, which now handle bad sectors; and PC99A.EXE, the accelerated version which had a minor bug uncovered when using Disk+Aid. These will all be incorporated in the next release of PC99.

Finally, we believe the process of uncovering SPAD's secrets shows that distinctions between the 4A world and the PC world tend to become blurred. The two machines are connected together with an umbilical RS232 cable which allows data to be transferred between them. If you need to look at the original disk you use Diags on the 4A. If you need to dump a file in hex it is quicker to use DUMP.EXE on the PC, and so on, allowing you to pick the best available tool to get the job done.

This lets you derive maximum benefit out of the hardware and software you own which, we think most people will agree, is a highly agreeable state of affairs.

MICRO-REVIEWS

PGRAM Utilities V2.3, Ian's Games, Schematic and Font Dumper

By CHARLES GOOD

PGRAM UTILITIES V2.3 by Tony Knerr

I reviewed this before, but it has been upgraded. You put the files on your PGRAM+ with multiple memory banks. These files in PGRAM+ banks 2, 3, and 4, plus XB v2.3 or some other extended Extended BASIC in bank 1 make a very powerful suite of software immediately available from the powerup menu.

TIWEA includes the TI-Writer and Editor/Assembler modules plus all the associated disk files all combined into one GROM bank. Included is Art Green's TIW v4.3 editor, which is much nicer than the original TIW. You can access this software from the powerup menu or by typing CALL EA or CALL TIW from either BASIC. The formatter and assembler return to their module's menu with Fctn/9.

(See Page 28)

MICROREVIEWS

(Continued from Page 27)

DSKUARCMC gives you Birdwell's DSKU, Barry Boone's Archiver, and Mike Dodd's M-Copier all combined into one GROM bank. CALL DSKU, CALL ARC, and CALL MCOPY will get you there from Extended BASIC. You can do the same thing by entering DSKU or ARC or MCOPY with no device name from the E/A5 loaders of the TIW and E/A modules.

GMENU will display all the software names, up to 24, of all your runnable software in all your GRAM banks simultaneously on one screen. GMENU is loaded onto the end of a GRAM page with other GRAM software.

You also get software that lets you use a PGRAM and a CorComp RAMdisk in the same P-box, something not normally possible. Finally, there is a version of the PGRAM DSR with colors set for white on dark blue rather than the anemic light blue found in the original.

PGRAM Utilities is public domain and comes on a DSSD disk. I'll send it to you for $1. The author requests an encouraging phone call, letter or CompuServe post as payment.

IAN'S GAMES
by Ian Howle

Ian Howle, author of "Attack of the Creepers" which I reviewed a few months ago, sent me a disk labeled "Ian's Games." In his cover letter he encourages me to distribute his assembly language games as widely as possible. They are public domain, he says, and based on computer games originally written in the late '70s. Ian's Games are in assembly language. I have seen similar not-as-good TI Extended BASIC versions of these games in the defunct International User Group software library. Each of Ian's games reviewed here comes with on-line instructions and an attractive title screen.

Tic Tac Toe. You get your choice of easy or hard levels and either one player against the computer or two players. I like this game because I never lose, even at the hard level. Tic Tac Toe is a game that can't be lost if you know the system, as I do. Even at the hard level, if the computer lets me start, which it does half the time, I can often win. Otherwise I get a tie game. You use a joystick to place your X or O on the board and press the fire button when the X or O is positioned with way you want. The screen display of the # pattern is quite artistic. The pattern is three-dimensional. You appear to be placing your X or O into three-dimensional boxes. This is the kind of visually attractive short game I really enjoy. When I get tired of doing other things I just boot up Ian's Tic Tac Toe and play a couple of quick games with the computer.

Seawolf. You are a submarine shooting straight up through the water at surface ships. These ships pass by on the surface from left to right or right to left at varying speeds, sometimes very fast. You have to sink a certain number of ships before time runs out. At successive levels you have to sink more ships in the same amount of time. There are rapidly moving fish and slowly moving mines in the water that can get in the way of your torpedoes and prevent them from reaching the surface. It is very annoying to have a fish zoom by and detonate a torpedo that you fired in an otherwise perfect setup. This game is hard! I rarely get beyond the first level. The fastest moving ships are almost impossible to sink.

Space Zap Deluxe. Your star base stays in the center of the screen and the invading hordes approach the star base one at a time from any of eight directions. You rotate your base's cannon in the appropriate direction with the joystick and shoot with the fire button to destroy the enemy coming from that particular direction. If the enemy gets too close to your base you are dead and the game starts over. The time interval between attacks gradually decreases. It takes a good joystick to work this game. Locking on to the proper diagonal is particularly difficult for most joysticks.

These three games plus a slightly updated version 1.6 of Attack of the Creepers all come on a SSSD disk which I'll mail you if you send me $1.

SCHEMATIC
by Don Steffen

Computers are supposed to do useful things, not just entertain or stimulate the mind. Schematic is an Extended BASIC program for a very specific application. It prints to a printer the schematic for a TI425 programmable controller or similar device made by TI, Allen-Bradley, and others. These devices are used control the operation of machines. They are, in effect, the robot brains of the machines they control.

The sample schematic printed by Schematic is the logic code of a Valley Hay Press. Don says these giant presses are made by his nephew. One is in Pennsylvania, two are in Australia and several are in the Pacific Northwest area. The presses are used to compress hay and straw for container shipment to Japan. Up to 24 tons can be put in a 40-foot container. The printed schematic of the machine's controller takes up several pages in condensed print. Each part of the schemati includes a specific name and number as well as a symbol. The author uses a Star NX1000 printer. I have no trouble getting a good printout on my old Star SG10. Both of these are nine-pin dot matrix printers.

Symbols are made of ASCII keyboard characters and are contained in DATA statements within the program. These symbols include such things as clock input, reset latch relay, master control end, ground terminal, hot terminal, indicator light, set latch relay, master control start, counter reset, solenoid coil, holding relay, etc. For example, the symbol for solenoid coil is "-(sol)-"; "-Reset |-" is for counter reset. A user could easily alter this symbolic notation to other specific symbols and meanings.

To get a printout you first, from within the program, make an elements file and add the symbols described above, their names and their specific numbers. This file is saved to disk and can be altered at any time. You then make a logic code file from within the program which is used by th program to format and print the schematic. This file is also saved to disk and can be

(See Page 29)

MICROREVIEWS —

(Continued from Page 28)

easily edited. Both these files are used by the program to make the printed schematic.

If you want a SSSD disk copy of Schematic along with the proper data files to print the schematic of the Valley Hay Press send the author a disk and paid return mailer, or send me $1. If you contact the author directly you might ask him about his perpetual calendar scheme, phonetic alphabet, and music notation characters, which are all very unusual and thought provoking and all of which are programmed on a 99/4A and displayed on screen and on a printer using a symbolic pattern based on the binary system.

FONT DUMPER
by Bruce Harrison

Here is another of the seemingly never-ending public domain contributions Bruce Harrison is making to the welfare of 99/4A users. You may remember that several years ago Jim Peterson released a whole bunch of public domain screen fonts to the public domain for use with Extended BASIC programs. There are 130 different fonts in the Peterson release! I have previously in this column reviewed a Harrison utility (Font Converter) that lets you convert these fonts to CHARA1 files for display on screen with your 40 or 80 column word processing documents. Now Bruce has taken the next step.

Font Dumper converts the Peterson screen fonts into a file that can be downloaded into the RAM of a NX1000 or NX1020 printer for use as a custom download NLQ printer font. This means that you can display the fonts on screen as CHARA1 files converted with Font Converter and then print the same fonts to your printer, all 130 of then if you want, as a WYSIWYG printout. Your Font Dumper disk contains full instructions and a couple of converted fonts. The conversion process has to be done with each font you want converted and it will take some time if you want to do all 130 fonts. You can preview the fonts from Extended BASIC using Jim Peterson's demo programs and than convert only those that turn you on.

Font Dumper is guaranteed to work with the NX1000 and NX1020 printers only. It may or may not work with other modern Epson-compatible printers. It does not work on my Gemini 10X or SG10 printers.

I can send you Font Dumper on a SSSD disk for either the NX1000 or NX1020 printers. There are different versions for each printer. Both versions means two disks. I can also send you on three DSSD disks the complete Jim Peterson screen font collection, with XB demo programs that display all the fonts, for you to convert with Font Dumper. Many TIers already have these screen fonts. I can also send you Font Converter on another SSSD disk so you can see these fonts on your TI-Writer screen. All these disks are public domain. Please send me $1 for each disk desired. Your money buys you the disks, mailer, and first class postage. Don't forget to tell Bruce Harrison how much you appreciate his efforts on our behalf.

ACCESS

Don Steffen (Schematic). 10082 Silverton Rd., Silverton, OR 97381. Phone (503) 873-4217.
Ian Howle (Ian's games). 3707 S.W. Southern St., Seattle, WA 98126. Phone (206) 938-4065
Tony Knerr (XBv2.3 and PGRAM Utilities author). 17 Marshall Circle, Downingtown, PA 19335. Phone (610) 269-7447. CompuServe #72070,573
Bruce Harrison (Font Dumper). Phone (310) 277-3467 9 a.m.-midnight.
Charles Good. P.O. Box 647, Venedocia, OH 45894. Phone (419) 667-3131. Internet email cgood@osulima1.lima.ohio-state.edu or good.6@osu.edu

First New England Fall Fair was a (small) success

By JACK SUGHRUE

The first New England Fall Fair, presented by the M.U.N.C.H. 99ers (Massachusetts Users of the Ninety-Nine Computer and Hobbyists) of Worcester, Massachusetts, officially christened a couple of major TI first.

During the morning sessions I had the honor of introducing Mickey Cendrowski's latest piece of software: Load Master. Originally conceived as Mouse Works, a loader program for the TI Mouse, it was changed during beta testing because so many people wanted keyboard access. Both mouse and keyboard code made the program too large and slow, so Mickey trimmed it down to operate as a fast keyboard loader with lots of exceptional qualities, including window layering (a la Funnelweb). In addition to providing exceptional loading properties, the program also supplies the user with a friendly method of changing defaults immediately and permanently (until the user intentionally writes over): screen and text colors, printer and so on. Many people at the fair went home with this shareware utility, which Mrs. Cendrowski says is still evolving. To secure this disk, contact your user group library or write her (I hope with a donation) at 100 Pine St., Russelton, PA 15076.

The second "first" for the M.U.N.C.H. event was a hardware one by Bud Mills, the developer of Horizon hardware peripherals. He had the SCSI (small computer systems interface) Card up and running on a normal TI. Though Bud explained that this was a limited version, it was the first public time this complicated interface card was operating. The audience was enthralled.

The audience was also caught up in the enthusiasm Charles Good (of Lima fame) showed toward the AMS (Asgard Memory Systems) card, which he demonstrated. The card seemed to operate flawlessly for about 10 minutes. Then each time Charles attempted to load TI Nopoly (which, along with other software, comes free with the card), the system locked up. Changes were

(See Page 30)

NEW ENGLAND FAIR —

(Continued from Page 29)

made: reseating cards, removing synthesizer, turning on fan. Nothing worked until a new console replaced the old. After that, all was perfect. With the 256K card he was able to demonstrate a variety of utilities and games. The demo was impressive.

As an aside, Charles proudly displayed his modified and strange-looking Geneve. It is the Porsche of the TI world and would require at least a page to describe in full.

Charles and I had a chance to tell the audience about the genius of Jim Peterson and the kind of man we found him to be. Interactive demonstration of his Nuts 'n Bolts Extended BASIC subprograms were also given.

Mike Wright and Mark Van Coppenolle of CaDD Electronics demonstrated their remarkable PC99. They captivated their audience for long past the hour set aside for the demo. At each question, Mike's fingers flew across the PC keyboard to show another way the TI sat perfectly inside the PC. Every TI utility and game mentioned by the viewers was immediately pulled to the screen for a demo. The Plato Library, Microsoft Multiplan, TI-Writer, all the education cartridges and all the game cartridges (many never seen without GRAMulators) were listed, displayed or discussed. Most of the men and women in the audience owned both computers and found the special features of PC99 intriguing.

Jim Cox, the M.U.N.C.H. exec in charge of this free event, said, "This is a big success. We were able to get a TI back in New England. Our fair is loaded with TI dignitaries and personalities: Bud, Charlie, Janet and Jennifer (Ryan), Mike, Mark, Barry (Traver) and all the others. It's a wonderful place to fraternize with the people who keep the TI going. We're all ready for next year."

In spite of the low turnout for the group's first event (47 adults, 4 children), the demonstrators and vendors agreed with Jim. At the end of the day, Charles Good summed it all up. "It was fun."

USER NOTES

Prescan tips can save time

The following item was written by Jim Swedlow and appeared in his column of Extended BASIC tips in ROM, the newsletter of the User Group of Orange County (California).

You load your program, enter RUN and

(See Page 31)

USER NOTES

(Continued from Page 30)

then ... nothing. Finally your program starts to execute. On a short program this wait is not noticeable, but on a long one it can seem endless. Why the delay?

Your 99/4A is going through your program line by line and alloting memory space. It is noting each variable used, each subprogram CALLed, the first DATA line, DEF statements, DIM statements, etc.

Furthermore, it is making an unduplicated list. Suppose you use the variable "A" 123 times in your program. The first time your 99/4A notes it and makes memory space. The other 122 times it checks, notes that it already knows about this variable and moves on. Even at the speed of the 4a, this takes time.

Extended BASIC has some tools to control prescan — !@P- turns it off and !@P+ turns it on. The following items must be in the range of the prescan:

- At least one use of each variable.
- At least one use of each CALL statement. For example, if you use CALL CLEAR five times, the first use must be within the prescan.
- All DEF, SUB, SUBEND, DIM and OPTION BASE statements.
- The first DATA statement in the program.
- !@P+ must be on a line by itself while !@P- can be at the end of a multi-statement line.

There is also a short-cut — prescan does look at CALL statements but it does not check validity. Therefore, you can do something like this:

```
10 DATA 2,3,4
20 OPTION BASE 1 :: DIM A(17
) :: GOTO 30 :: CALL HCHAR ::
 :: CALL SPRITE :: CALL SAY :
: R,S,T,U,V=W :: A$=B$ :: !@
P-
30 ! Program continues
```

Note that the code after GOTO 30 in line 20 will never be executed, so it does not need to meet syntax requirements.

You should not activate prescan until your program is fully debugged. If you forget something, you will get a syntax error.

With a bit of work, you will cut the prescan time down significantly.

This is from the documentation for Neatlist: CALL PEEK(8198,A) will let you know if a CALL INIT has been performed. If "A" returns as 170, then it has, any other value indicates that it has not.

UNITED STATES POSTAL SERVICE™

Statement of Ownership, Management, and Circulation
(Required by 39 U.S.C. 3685)

1. Publication Title	2. Publication No.	3. Filing Date
MICROpendium	1 0 4 3 - 2 2 9 9	10-1-95
4. Issue Frequency: monthly	5. No. of Issues Published Annually: 12	6. Annual Subscription Price: $35/yr

7. Complete Mailing Address of Known Office of Publication (Street, City, County, State, and ZIP+4) (Not Printer)
502 Windsor Rd., Round Rock, Williamson, TX 78664-7639

8. Complete Mailing Address of Headquarters or General Business Office of Publisher (Not Printer)
P.O. Box 1343, Round Rock, TX 78680-1343

9. Full Names and Complete Mailing Addresses of Publisher, Editor, and Managing Editor (Do Not Leave Blank)
Publisher (Name and Complete Mailing Address)
John Koloen, P.O. Box 1343, Round Rock, TX 78680-1343

Editor (Name and Complete Mailing Address)
Laura Burns, P.O. Box 1343, Round Rock, TX 78680-1343

Managing Editor (Name and Complete Mailing Address)
NA

10. Owner (If owned by a corporation, its name and address must be stated and also immediately thereafter the names and addresses of stockholders owning or holding 1 percent or more of the total amount of stock. If not owned by a corporation, the names and addresses of the individual owners must be given. If owned by a partnership or other unincorporated firm, its name and address as well as that of each individual must be given. If the publication is published by a nonprofit organization, its name and address must be stated.) (Do Not Leave Blank.)

Full Name	Complete Mailing Address
Burns-Koloen Communications Inc.	P.O. Box 1343, Round Rock, TX 78680-1343
John Koloen	P.O. Box 1343, Round Rock, TX 78680-1343
Laura Burns	P.O. Box 1343, Round Rock, TX 78680-1343

11. Known Bondholders, Mortgagees, and Other Security Holders Owning or Holding 1 Percent or More of Total Amount of Bonds, Mortgages, or Other Securities. If none, check here. ☐ None

Full Name	Complete Mailing Address

12. For completion by nonprofit organizations authorized to mail at special rates. The purpose, function, and nonprofit status of this organization and the exempt status for federal income tax purposes: (Check one)
☐ Has Not Changed During Preceding 12 Months
☐ Has Changed During Preceding 12 Months
(If changed, publisher must submit explanation of change with this statement)

PS Form 3526, October 1994 (See Instructions on Reverse)

13. Publication Name: MICROpendium	14. Issue Date for Circulation Data Below: September 1995	
15. Extent and Nature of Circulation	Average No. Copies Each Issue During Preceding 12 Months	Actual No. Copies of Single Issue Published Nearest to Filing Date
a. Total No. Copies (Net Press Run)	1458	1300
b. Paid and/or Requested Circulation (1) Sales Through Dealers and Carriers, Street Vendors, and Counter Sales (Not Mailed)	206	172
(2) Paid or Requested Mail Subscriptions (Include Advertisers' Proof Copies/Exchange Copies)	782	694
c. Total Paid and/or Requested Circulation (Sum of 15b(1) and 15b(2))	988	866
d. Free Distribution by Mail (Samples, Complimentary, and Other Free)	52	48
e. Free Distribution Outside the Mail (Carriers or Other Means)	10	0
f. Total Free Distribution (Sum of 15d and 15e)	62	48
g. Total Distribution (Sum of 15c and 15f)	1050	914
h. Copies Not Distributed (1) Office Use, Leftovers, Spoiled	306	386
(2) Return from News Agents	2	0
i. Total (Sum of 15g, 15h(1), and 15h(2))	1458	1300
Percent Paid and/or Requested Circulation (15c / 15g x 100)	94%	95%

16. This Statement of Ownership will be printed in the October issue of this publication. ☐ Check box if not required to publish.

17. Signature and Title of Editor, Publisher, Business Manager, or Owner — Date
Laura Burns, Editor — Oct. 1, 1995

I certify that all information furnished on this form is true and complete. I understand that anyone who furnishes false or misleading information on this form or who omits material or information requested on the form may be subject to criminal sanctions (including fines and imprisonment) and/or civil sanctions (including multiple damages and civil penalties).

Instructions to Publishers

1. Complete and file one copy of this form with your postmaster on or before October 1, annually. Keep a copy of the completed form for your records.
2. Include in items 10 and 11, in cases where the stockholder or security holder is a trustee, the name of the person or corporation for whom the trustee is acting. Also include the names and addresses of individuals who are stockholders who own or hold 1 percent or more of the total amount of bonds, mortgages, or other securities of the publishing corporation. In item 11, if none, check box. Use blank sheets if more space is required.
3. Be sure to furnish all information called for in item 15, regarding circulation. Free circulation must be shown in items 15d, e, and f.
4. If the publication had second-class authorization as a general or requester publication, this Statement of Ownership, Management, and Circulation must be published; it must be printed in any issue in October or the first printed issue after October. If the publication is not published during October.
5. In item 16, indicate date of the issue in which this Statement of Ownership will be printed.
6. Item 17 must be signed.

Failure to file or publish a statement of ownership may lead to suspension of second-class authorization.

PS Form 3526, October 1994 (Reverse)

The ONLY monthly devoted to the TI99/4A

Subscription Fees

❑ 12 issues, USA, **$35**

❑ 12 issues, Mexico, **$40.25**

❑ 12 issues, Canada **$42.50**

❑ 12 issues, other countries surface mail, **$40.00**

❑ 12 issues, other countries, air mail, **$52.00**

Outside U.S., pay via postal or international money order or credit card; personal checks from non-U.S. banks will be returned.

Address Changes

Subscribers who move may have the delivery of their most recent issue(s) delayed unless MICROpendium is notified six weeks in advance of address changes. Please include your old address as it appears on your mailing label when making an address change.

Check each item ordered (or list on separate page) and enter total amount here:_____

Check/MO ❑ MasterCard ❑ VISA ❑ (check one)

Card No. _____

Expiration Date _____

(Minimum credit card order is $9)

Signature _____

(Required on credit card orders.)

No sales tax on magazine subscriptions. Texas residents add 7.75% sales tax on other items, including back issues and disk subscriptions. Credit card orders add 5%.

Mail to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680

Name _____

Address _____

City _____

State _____ **ZIP** _____

The set of numbers at the top of your mailing label indicates the cover date of the last issue of your subscription.

Disks, Etc.

❑ **Back Issues**, $3.50 each. List issues:_____

_____

No price breaks on sets of back issues. Free shipping USA. Add 30 cents, single issues to Canada/Mexico. Other foreign shipping 50 cents single issue surface, $1.50 airmail. Write for foreign shipping on multiple copies.

OUT OF STOCK: Vols. 1, No. 1-2; Vol. 2, No. 1

❑ **MICROpendium Index** (2 SSSD disks, 1984-1992), Extended BASIC required $6.00

❑ **MICROpendium Index II** (9 SSSD disks — 1 for each year — 1984-1992), XB required $30.00

❑ **MICROpendium Index II with MICROdex 99** (11 SSSD disks), XB required.....$35.00

❑ **MICROdex 99** (for use with MP Index II, 2 SSSD disks), XB required $10.00

❑ **MICROpendium Index II** annual disks ordered separately (1 disk per year, 1984-1992); each $6.00

MICROdex 99, by Bill Gaskill, is a collection of programs that allow users of MP Index II to modify their index entries, as well as add entries. MICROdex 99 supports many other functions, including file merging, deletion of purged records, record counting and file browsing.

GENEVE DISKS (SSSD unless specified)

❑ **MDOS 2.21** (req. DSSD or larger (for floppy & hard drive systems) $4.00

❑ **GPL 1.5** $4.00

❑ **Myarc Disk Manager 1.50**.....$4.00

❑ **Myarc BASIC 3.0**.....$4.00

❑ **MY-Word V1.21** $4.00

❑ **Menu 80** (specify floppy or hard disk versions(s); includes SETCOLR, SHOWCOLOR, FIND, XUTILS, REMIND $4.00

GENEVE PUBLIC DOMAIN DISKS

These disks consists of public domain programs available from bulletin boards. If ordering DSDD, specify whether Myarc or CorComp.

	SSSD	DSSD	DSDD
❑ **Series 1**	$9.00	$7.00	$5.00
❑ **Series 2**	$9.00	$7.00	$5.00
❑ **Series 3**	$9.00	$7.00	$5.00
❑ **Series 4**	$9.00	$7.00	$5.00
❑ **Series 5**	$9.00	$7.00	$5.00
❑ **Series 6**	$9.00	$7.00	$5.00

SECOND CLASS