$2.50

REMark®

Volume 6, Issue 3 • March 1985

P/N 885-2062

Official magazine for users of HEATH ZENITH computer equipment.

High Powered Ammunition

The pros at First Capitol Computer know what you want—high powered ammunition for your Zenith. So we've engineered a series of the highest quality hard drives available for your Zenith 150 PC.

If you're a power PC user, and only have a floppy based system, you're probably wishing for the power of a fast hard disk. Or maybe you have the standard 11 Mb drive and want something larger. Like 36 Mb.

Use the full capabilities of powerful programs like Ashton Tate's D-Base III™ and Framework™; Lotus 1-2-3™ and Symphony™, and Micropro Wordstar™.

A Full Range Of Calibres

Our drive kits come in a size for every need; the standard 11 Mb, a serious 20 Mb, and a full bore 36 Mb.*

Our drives will allow autoboot from the hard disk, and require no permanent modifications to your existing hardware, and no changes to your operating system software—we've got everything needed in ROM on the controller board, customized for the Zenith PC.

Technical Wizardry

Our drive kits are engineered by Software Wizardry, the Zenith experts the experts consult, and are available exclusively through First Capitol Computer. You can count on leading-edge performance; quiet, smooth, and fast operation that will speed you through your most serious software.

All Hard Disks Are Not Created Equal

A lot of people are selling hard disks at a lot of prices—know what you are getting! Rapidly improving technology has resulted in the dumping of obsolete drives on the market. Not at First Capitol Computer!

Our drive kits use the latest state-of-the-art drives, with the newest media, head designs, fast access time, and low power consumption. You can be assured of the highest quality, equal to Zenith's original manufacturing standards.

Firepower

Load A High Calibre Winchester In Your Zenith

Value Priced

Our prices are some of the most competitive around, and when you consider our latest state-of-the-art components, First Capitol's hard disk upgrades are the best value on the market for your Z-150 PC series.

All You Need To Know

You don't have to be a technical wizard to install our systems—our wizards have done that for you! You get the drive, controller card, all necessary cables, and complete instructions.

Available Models:	List Price
SWI-WIN150-11 11 megabyte* internal hard disk kit	$ 895.00
SWI-WIN150-20 20 megabyte* internal hard disk kit	$1195.00
SWI-WIN150-36 36 megabyte* internal hard disk kit	$1595.00

Available direct from First Capitol Computer. Please add $2 minimum (or 2%, whichever is greater) for shipping and handling. If shipped to a Missouri address, please add appropriate sales tax.

*drive sizes are unformatted capacities

First
Capitol
Computer

First Capitol Computer is a division of Software Wizardry, Inc.

1106 First Capitol Drive
St. Charles, MO 63301

Express Order Line (orders only please)
1-(800)-TO-BUY-IT (1-800-862-8948)

(314) 946-1968 (technical info)

Finally, text processing that fits your computer to the letter.

Newline. Matched to the Zenith 100, 150 and the IBM-PC. If you have one of these systems, Newline has the right line of text processing software for you. Because Newline's Professional Text Processor (PTP) is matched to the keyboard and display characteristics of each of these computers.

There are no complicated key sequences to learn. And you'll be able to use labeled editing keys. Which means it's exceptionally easy to use.

New features for Newline's PTP. Even better, now PTP has more powerful text processing than ever before. There's everything from full screen text editing and on-screen bold, underline, paragraph fill and justification to cut and paste to configurable macro keys and much more. Plus, you'll be able to use our software with any printer.

Update from our old line. If you're presently using the Newline TxtPro software on your Zenith 100, now you can update to our more powerful version. It's called the PTP-100. If you currently have the ZDOS TxtPro, you can upgrade to ZDOS PTP-100. If you have the CP/M-85 TxtPro, you can upgrade to CP/M-86 PTP-100, but you'll also need to upgrade your system to CP/M-86. Just specify which one you have when you order. And if you return your old TxtPro disk to us with your order, you qualify for a special reduced price.

For Zenith 150 and IBM-PC users, there's the new PTP-PC. And it has all the same features as the PTP-100.

Software development editing. For programmers, Newline's PTP also offers auto indent and produces ASCII files for use with assemblers, interpreters and compilers. And that's not just different, it's unique.

Just give us the word. Newline's Professional Text Processor (PTP) is available right now. So place your order today. And get the text processing software that fits *your* system to the letter.

NAME ______________________________

ADDRESS ______________________________

CITY ______________ STATE _____ ZIP ________

PTP-100 (for Z100) @ **$99. or**
PTP-100 (Z100 Upgrade) @ **$50.***

	Qty	Amount
□ CP/M-86 (replaces CP/M-85)	____	____
□ ZDOS	____	____

*To qualify for Z100 upgrade price of $50, you must return your TxtPro disk.

PTP-PC (for Z150 or IBM-PC) @ **Special Introductory Price of $149.**

	Qty	Amount
□ CP/M-86	____	____
□ MS-DOS or PC-DOS	____	____
RI Residents Add 6% Sales Tax		____
Shipping ($3. per program)		____
TOTAL ENCLOSED		____

Payment in U.S. Funds Only • Allow 2 wks. for Personal Checks • CALL (401) 624-3322 FOR C.O.D. DELIVERIES

Newline Software
P.O. Box 289 • Tiverton, RI 02878 • (401) 624-3322

Heath/Zenith
Users'
Group

Staff

Manager .... Bob Ellerton
(616) 982-3867

Software Engineer .... Pat Swayne
(616) 982-3463

Bulletin Board and
Software Developer ..... Jim Buszkiewicz
(616) 982-3463

Software Coordinator .... Nancy Strunk
(616) 982-3838

Secretary .... Margaret Bacon
(616) 982-3463

REMark

Editor .... Walt Gillespie
(616) 982-3789

Editorial and Advertising
Assistant .... Lori Latham
(616) 982-3794

Printers .... Imperial Printing
St. Joseph, MI

	U.S. Domestic	APO/FPO & All Others
Initial	$20.00	$35.00*
Renewal	$17.00	$30.00*

* U.S. Funds

Limited back issues are available at $2.50, plus 10% shipping and handling. Check HUG Product List for availability of bound volumes of past issues. Requests for magazines mailed to foreign countries should specify mailing method and appropriate added cost.

Send Payment to: Heath/Zenith Users' Group
Hilltop Road
St. Joseph, MI 49085
(616) 982-3463

Although it is a policy to check material placed in REMark for accuracy, HUG offers no warranty, either expressed or implied, and is not responsible for any losses due to the use of any material in this magazine.

Articles submitted by users and published in REMark, which describe hardware modifications, are not supported by Heathkit Electronic Centers or Heath Technical Consultation.

HUG is provided as a service to its members for the purpose of fostering the exchange of ideas to enhance their usage of Heath equipment. As such, little or no evaluation of the programs or products advertised in REMark, the Software Catalog, or other HUG publications is performed by Heath Company, in general and HUG, in particular. The prospective user is hereby put on notice that the programs may contain faults, the consequence of which Heath Company, in general, and HUG, in particular, cannot be held responsible. The prospective user is, by virtue of obtaining and using these programs, assuming full risk for all consequences.

REMark®

Official magazine for users of HEATH ZENITH computer equipment.

Volume 6, Issue 3 • March 1985

Index of Advertisers

This index is provided as an additional service. The publisher does not assume any liability for errors or omissions.

On The Cover: *Shown is the Printek 930 printer. Is it the ultimate in dot matrix? Story on Page 46. (Photo by Jory Klopp, Heath Graphic Design)*

IN 1979, DISCOVERY
HARD DISK YOU COULD

IN 1980, DITTO.

IN 1981-82, DITTO.

Back in 1979, we at ACT built the best 5¼″ hard disk subsystems for your Zenith. Today the same holds true. For the same reasons.

DISCOVERY hard disk subsystems still offer you the finest on-line storage devices. Capacities that run from 5 to 225 formatted megabytes mean we can take on your toughest business applications and largest database needs and still have room to spare! Our 5- and 10-megabyte removable hard disks offer you all the advantages of true random access media together with all the benefits of storing your sensitive data at separate, secure locations.

And don't forget we pioneered and developed our original digital data separator and controller right on a Zenith system. The results? One of the fastest, most reliable

BUGGIN' HUG

Never-Neverland

Dear HUG:

While working on a device driver for my Z-100, I found an error in the MS-DOS Version 2 Programmer's Utility Pack manual, part number 595-3186-1. On page 8.2 under the discussion of passing parameters to IO.SYS, the manual claims that as "a rule, IO.SYS routines preserve all registers except for AX and the flags." I found that in the case of CALLS to the BIOS_CONFUNC, BIOS_AUXFUNC, and BIOS_PRNFUNC functions, some of the other registers are, in fact, modified. The comments in my MS-DOS 2.13 BIOS source code indicate that BX, DX, DI, and SI are modified. If your calling program fails to preserve these registers before CALLing IO.SYS, it is likely to go to never-neverland.

Gary R. Cramblitt
HQ 5th Sig Comd
P.O. Box 138
APO NY 09056

Closed To Me Until Problem Is Solved

Dear Walt:

I have been successfully using a Zenith Z-90 with Z-37 Disk Drive since June of 1983. At the time of purchase of the system (new), I was able to pick up a C-ITOH Starwriter (used) for a very good price. The catch was that the Starwriter was parallel and all I had was serial ports. As fate would have it, at very nearly the same moment, Heath was introducing a new serial/parallel interface board which "would solve my problems" with regard to installing the printer. True (to a point) and although substantial rework on the BIOS was necessary by the dealer who sold me the system, the printer has been up and running in solid fashion with no hitches during the past two years.

The problem which I am concerned about may or may not have anything to do with this special board, but I have always suspected it. I literally received the very first copy of this animal (I don't even know the name or number of it and my dealer has since gone out of business) and it was supposed to have included some sort of software patch which was intended to make all of the appropriate bells and whistles work. The general operation of the computer has never really been in question, at least from the standpoint of this novice "operator," however, as you might well understand, as I have become more and more experienced in using this trusty piece of hardware, I have found an increasing need to be able to attach a modem . . . which brings me to my problem. I cannot get this machine to recognize a modem. I have tried to talk to the local Heath repair people about it but they seem substantially less than anxious to get involved. I know that there are many things which would probably be of great use to me which are floating around out there on the various HUG Bulletin Boards, however, they are closed to me unless I can solve this problem. I have a couple of different modem programs available to me, including the CPS program which is certainly not exceedingly complicated, but none of them will perform under the current circumstances.

If there is anybody out there who might have even a vague idea of where my problem might lie, I would certainly like to hear from them. I know that there has to be a way. I don't know if the custom BIOS, which was written by my dealer, might be part of the problem or not, but I suspect anything at this poiint. My main problem is that I am really an operator and not a programmer or even a qualified student of the hardware environment, which puts me at a substantial disadvantage when it comes to assessing problems such as these.

Hoping you can help!

Rodger A. Gray
5205 Edgemont Circle
Cypress, CA 90630

A Little Late For Chanukah

Dear HUG:

This program is for the H/Z-100, ZDOS, and ZBASIC. It creates a Jewish manorah with the following colors: yellow, red, magenta, green, and blue. It may be a little late for Hanukkah (or Chanukah), but try this program anyway. The reason I contributed this program to Buggin' HUG is because I've used a-LOT of REMark programs and I figured it was my time to contribute. The listing follows.

Sincerely,

Jonathan Burnstein
632 So. Lincoln Street
Hinsdale, IL 60521

```
10 'CHANUKAH.BAS BY JONATHAN BURNSTEIN. MAKES A JEWISH MANORAH
20  COLOR 15,0: SCREEN 0: WIDTH 80: KEY OFF: CLS
30 'DRAW AND PAINT OUTLINE
40 LINE (219,210) - (283,196),6:
   LINE (283,196) - (283,144),6
50 LINE (219,210) - (384,210),6:
   LINE (384,210) - (320,196),6
60 LINE (320,196) - (320,144),6:
   LINE (320,144) - (545,144),6
70 LINE (282,144) - (45,144),6:
   LINE (45,144) - (42,142),6
80 LINE (42,142) - (38,139),6:
   LINE (38,139) - (38,134),6
90 LINE (38,134) - (38,108),6:
   LINE (38,108) - (64,90),6,B
100 LINE (64,90) - (64,124),6: CIRCLE (79,126),15,6,2.9
110 LINE (95,126) - (95,108),6:
    LINE (95,108) - (121,90),6,B
120 LINE (121,90) - (121,124),6: CIRCLE (136,126),15,6,2.9
130 LINE (152,90) - (177,108),6,B:
    LINE (152,90) - (152,125),6
140 LINE (177,108) - (177,124),6: CIRCLE (192,126),15,6,2.9
150 LINE (207,108) - (207,126),6:
    LINE (207,108) - (232,90),6,B
160 LINE (232,108) - (232,127),6: CIRCLE (258,126),26,6,2.9:
    PSET (233,123),0
170 LINE (284,125) - (284,81),6:
    LINE (284,42) - (320,81),6,B
180 LINE (320,81) - (320,125),6: CIRCLE (346,125),26,6,2.9:
    PSET (321,122),0
190 LINE (372,124) - (372,108),6:
    LINE (372,108) - (397,90),6,B
```

```
200 LINE (397,126) - (397,108),6: CIRCLE (412,126),15,6,2.9
210 LINE (427,126) - (427,108),6:
    LINE (427,90) - (450,108),6,B
220 LINE (450,126) - (450,108),6: CIRCLE (465,126),15,6,2.9
230 LINE (480,126) - (480,108),6:
    LINE (480,108) - (502,90),6,B
240 LINE (502,108) - (502,126),6
250 CIRCLE (517,126),15,6,2.9: LINE (532,108) - (532,126),6
260 LINE (532,108) - (557,90),6,B:
    LINE (557,108) - (557,140),6
270 LINE (546,144) - (557,140),6:
280 'SPELLING "HAPPY CHANUKAH!!"
290 LINE (36,9) - (36,36),5: LINE (36,23) - (64,23),5
300 LINE (64,36) - (64,9),5: LINE (82,36) - (98,8),5
310 LINE (98,8) - (113,36),5: LINE (90,23) - (106,23),5
320 LINE (134,9) - (134,36),5: CIRCLE (136,16),18,5
330 LINE (133,6) - (115,36),0,BF: LINE (179,9) - (179,36),5
340 CIRCLE (181,16),18,5: LINE (178,6) - (162,36),0,BF
350 LINE (214,9) - (227,20),5: LINE (242,9) - (227,20),5
360 LINE (227,20) - (227,36),5: CIRCLE (354,25),30,2
370 LINE (374,0) - (424,36),0,BF:
    LINE (397,9) - (397,36),2
380 LINE (423,9) - (423,36),2: LINE (397,23) - (423,23),2
390 LINE (448,36) - (464,9),2: LINE (464,9) - (480,36),2
400 LINE (456,23) - (472,23),2: LINE (497,9) - (497,36),2
410 LINE (497,9) - (520,36),2: LINE (520,36) - (520,9),2
420 CIRCLE (550,5),30,2,2,,3: LINE (480,0) - (639,8),0,BF
430 LINE (577,23) - (600,23),2:
    LINE (336,153) - (336,180),2
440 LINE (336,165) - (365,153),2:
    LINE (346,162) - (368,180),2
450 LINE (384,180) - (400,153),2:
    LINE (400,153) - (416,180),2
460 LINE (392,167) - (408,167),2:
    LINE (432,153) - (432,180),2
470 LINE (460,153) - (460,180),2:
    LINE (432,167) - (460,167),2
480 LINE (480,153) - (480,172),2: CIRCLE (480,180),5,2:
    PAINT (482,180),2,2
490 LINE (496,153) - (496,172),2: CIRCLE (496,180),5,2:
    PAINT (498,180),2,2
500 'THE 2 FLAMES
510 CIRCLE (52,82),13,6: PAINT (52,82),6,6:
    LINE (40,79) - (52,62),6
520 LINE (52,62) - (64,79),6: PAINT (52,69),6,6:
    CIRCLE (301,29),16,6
530 PAINT (301,29),6,6: LINE (286,25) - (301,9),6:
    LINE (301,9) - (314,25),6
540 PAINT (302,15),6,6: PSET (288,24),6
550 'BY...
560 LOCATE 25: PRINT "by Jonathan burnstein";
570 IF INKEY$ = "" THEN 570
580 END
```

A Bug And An Enhancement

Dear HUG:

This letter describes a bug in the sorted directory utility, ZD (REMark, April 1984), and an enhancement to the program. First, the bug: ZD crashes if it attempts to list a directory with one file. The fix is simple, 2 instructions in the Sort procedure, as shown in Listing 1 (the added lines are marked with a "%" in the comment field).

It might seem unlikely that one would want to list a directory containing only one file. However, the situation arose while I was modifying ZD to accept a wildcard file specification in the style of the ZDOS command, dir, rather than always listing the entire directory. The modifications necessary for this are shown in Listing 2. The syntax of ZD is then exactly the same as that of dir. Note that there is a change in the original syntax of ZD, i.e. each option switch must be preceded by "/" and it is not necessary to precede the "/" by a space. This is to conform to the ZDOS v 1.25 conventions. I hope that this modification is useful to other REMark readers.

Sincerely yours,

Timothy A. Gonsalves
53-A Escondido Village
Stanford, CA 94305

Listing 1

Bug fix in procedure Sort

```
Sort     Proc    Near
         Mov     Stkcnt,CX         ;Store number of entries
         Dec     CX                ;
         jg      DoSort            ;% sort only if >= 2 files

         ret                       ;%

DoSort:  Mov     SI,Offset pntr    ;% SI is first file entry
                                   ; ptr
Outer:   Mov     DI,SI             ;
```

Listing 2

Modification to ZD to allow wildcard specifications

(Again, added or modified lines have a "%" in the comment field).

```
Cknxt:    Inc     Bx                ;
          Dec     Cl                ;Search for the option
                                    ;delimiter
          Js      CkDrv
          Cmp     Byte ptr [Bx],'/'
          Jnz     Cknxt             ;% read one flag per '/'

          Inc     Bx                ;% Here only on valid '/'
          Dec     Cl                ;Check option field for
          Js      CkDrv             ;  legal options

...

Note: Delete label CkOpt:; Replace all jump to CkOpt
      by jump to CkNxt.

...

CkDrv:                              ;% look for filespec, parse
                                     if found
          mov     si, 80h           ;% Point to cmd buffer
          mov     cl, byte ptr [si]  ;% Get length of buffer
          xor     ch, ch

DvNxt:    inc     si                ;%
          dec     cl
          js      DfDrv             ; Use default if nothing
                                     entered

          mov     al, byte ptr [si] ;% get next char
          cmp     al, ' '           ;% non-space?
          je      DvNxt

          cmp     al, '/'           ;% is it a flag?
          jne     GetFileSpec       ;%

          inc     si                ;% skip flag
          dec     cl                ;%
          jmp     DvNxt             ;%

GetFileSpec:                        ;% parse filespec pointed to
                                     by ds:si
          push    ds                ;%
          pop     es                ;% for parse output, es:di
          mov     di, offset Dfcb ;% place for filespec
          mov     al, 1110b         ;% dont change a field if
                                     it is omitted
```

```
         mov     ah, 29h          ;% DosF_Parse
         int     21h              ;%

         mov     al, Dfcb         ;% get drive no.
         cmp     al, 0            ;%
         je      DfDrv            ;% do nothing if default

         add     al, 'A'-1        ;% convert to Ascii
         mov     cDrv, al         ;% save for display

         mov     al, Dfcb         ;%
         cmp     al, 0            ;% Check is valid drive
         jge     Drvok            ;% branch if it looks ok

Errl:    Mov     Dx, offset Ermsg1

Note: Some lines between labels DvNxt: and Err1: have
      been deleted.
```

"My Favorite Subroutines" Or Buggin' HUG?

Dear HUG:

It is sort of a toss-up whether this should be addressed to the Buggin' HUG feature, or to "My Favorite Subroutines," for Jeff Kalis' sorted directory program has just got to be one of my favorites. However, I have found that if the routine attempts to sort a list of length 1, it gets hung up in an endless loop. I discovered this when I created a file on Ramdrive, then forgot what I had named it, and used the routine to find the name.

In the original assembly language as published in the April '84 REMark, the sort procedure was as given below. The comments are for the case of an initial list of length 1.

```
SORT     PROC    NEAR
         MOV     STKCNT, CX         ;CX has length of list
         DEC     CX                 ;If CX was 1, now zero
         MOV     SI, OFFSET PNTR
OUTER:   MOV     DI, SI
         ADD     DI, 2
         MOV     DX, CX             ;DX now zero, same as CX
INNER:   CALL    COMPAR
         JBE     LEAVE
         MOV     AX, WORD PTR [SI]
         XCHG    AX, WORD PTR [DI]
         MOV     WORD PTR [SI], AX
LEAVE:   ADD     DI, 2
         DEC     DL                 ;If CX is 0, DL is now
         JNZ     INNER              ;-1, and will never let
         ADD     SI, 2              ;us out of the loop.
         LOOP    OUTER
         RET
SORT     ENDP
```

With a slight rearrangement, as given below, the hang-up will not occur.

```
SORT     PROC    NEAR
         MOV     STKCNT, CX
         MOV     SI, OFFSET PNTR
         MOV     DX, CX
OUTER:   DEC     DX
         JZ      DEPART             ;No more swaps needed if
         MOV     CX, DX             ;DX = 1 on arrival at
         MOV     DI, SI             ;OUTER
         ADD     DI, 2
INNER:   CALL    COMPAR
         JBE     LEAVE
         MOV     AX, WORD PTR [SI]
         XCHG    AX, WORD PTR [DI]
         MOV     WORD PTR [SI], AX
LEAVE:   ADD     DI, 2
         LOOP    INNER
         ADD     SI, 2
         JMP     OUTER
DEPART:  RET
SORT     ENDP
```

I hope this will help increase the well deserved popularity of the program. I am so attached to it that I have it on almost every disc I use.

Sincerely,

Robert G. Brasfield
303 N. 175th Street
Seattle, WA 98133

Morse Code Amended

Dear HUG:

I have amended my copy of Robert Horn's Morse Code program (REMark 41, page 33) to use graphic characters, single character and tutorial capabilities. The original capabilities of the program are still in tact, with the exception of changing the five-letter word groups from single line size to 4-character-high letters (Jumbo Letters, REMark 42, by David Warnick). I found Robert Horn's program very useful in learning the Morse Code, but it was hard to read the small size word groups. Plus there was no capability to get a user response, as I have added in the tutorial section. To include the above features you will have to change or add the following as indicated.

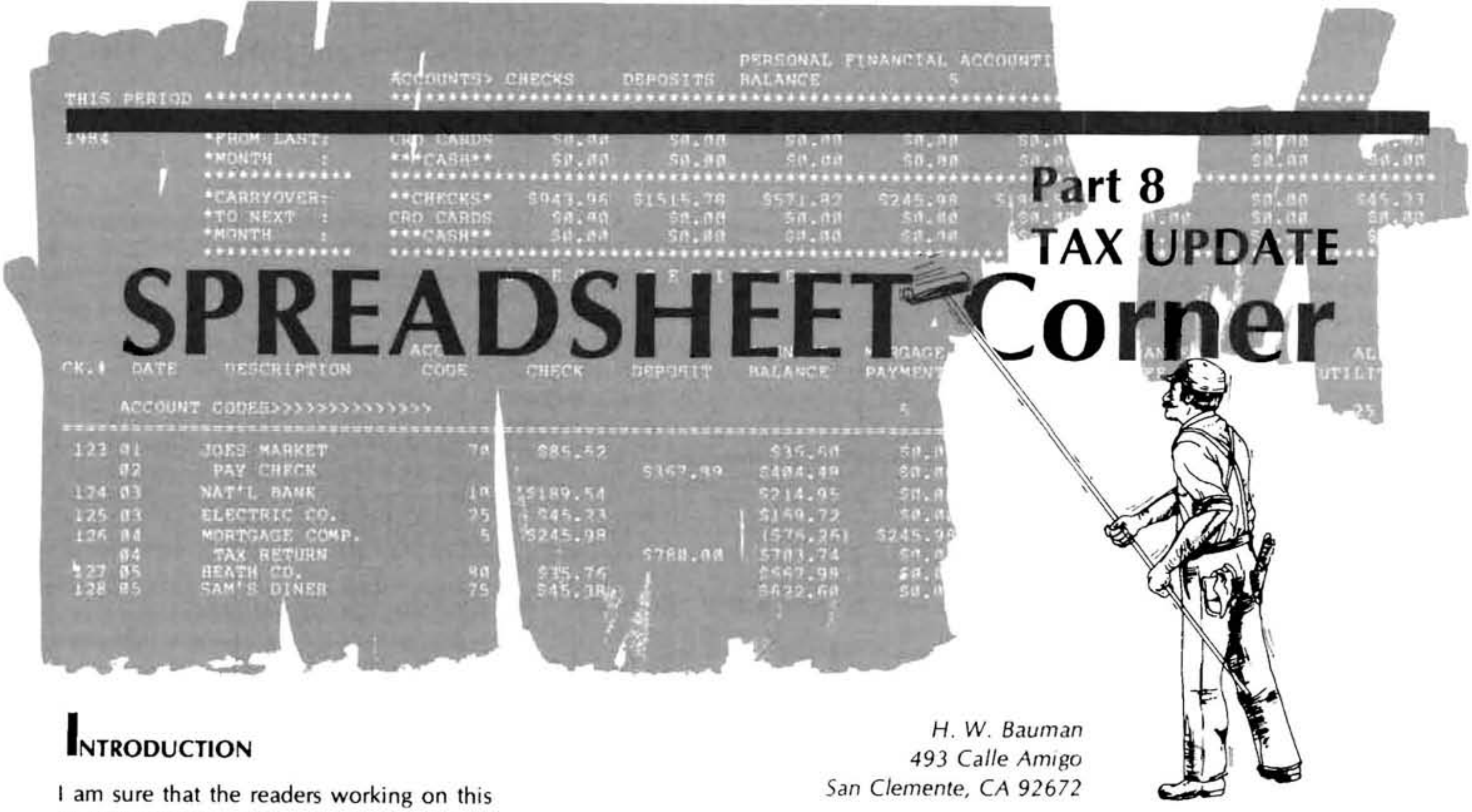
Change lines xxx to read:

```
270 IF CH$="P" OR CH$="p" THEN PRINT " --> Practice Code <--":
    GOTO 545:ELSE PRINT " —> Convert Text File to Code <--"
370 IF PC$="Y" OR CH$="N" THEN PRINT FNCC$(1,21);
    K;FNCC$(2,1);EE$;:GOTO 4000
460 FOR T=1 TO N3:NEXT T
550 PRINT:INPUT "How Many Five-Letter Words (Max. 100) ";
    N5
```

Insert lines xxx as follows:

```
52 CA$=E$+"Y":SH$=E$+"[":SS$=E$+"\":GM$=E$+"F":
   GO$=E$+"F":GO$=E$+"G":ED$=E$+"E"
54 VR$=E$+"p":VN$=E$+"q":CX$=E$+"x5":CO$=E$+"y5":
   EE$=E$+"J"
56 DEF FNCC$(X,Y)=CA$+CHR$(X+31)+CHR$(Y+31)
372 IF CH$="Y" THEN PRINT CX$;ED$:GOTO 380
375 PRINT CX$;L$(K)
461 IF CH$<>"Y" GOTO 468
462 PRINT ED$;CO$:PRINT FNCC$(12,31);
    "Input Your Guess?  ";:GU$=INPUT(1):PRINT GU$
464 IF GU$=L$(K) THEN PRINT FNCC$(14,37);"Correct":
    C=C+1
466 IF GU$<>L$(K) THEN PRINT FNCC$(14,27);
    "Wrong.  Correct Answer is ";L$(K):
    PRINT CX$:FOR T=1 TO 500:NEXT T
468 NEXT K
469 IF CH$="Y" THEN TC=INT((C/N5)*100+.5):
    PRINT FNCC$(18,26);"You Answered ";TC;"% Correctly"
545 PRINT:PRINT
    "Practice Random <A>lphabet or 5 Letter <W>ord Groups?
    <W>";:CH$=INPUT(1)
546 IF CH$="A" OR CH$="a" THEN PRINT " --> Alphabet <--"
    ELSE PRINT "—> Words <--"
547 IF CH$="A" OR CH$="a" GOTO 3000
3000 PRINT:INPUT
     "How Many Random Alpha Characters (Max. 100) ";N5
3010 IF N5>100 GOTO 3000
3020 PRINT:W=1
3030 FOR I=1 TO N5
```

Vectored to Page 65 ☞



SPREADSHEET Corner

Part 8
TAX UPDATE

H. W. Bauman
493 Calle Amigo
San Clemente, CA 92672

INTRODUCTION

I am sure that the readers working on this project are aware of the numerous changes that resulted from the "Tax Reform Act of 1984" that was signed into law by President Reagan on July 18, 1984. These changes were primarily enacted to increase tax revenues in a continuing effort to reduce the national deficit. The changes substantiality effected some existing tax provisions and postponed and/or repealed other tax laws that were enacted in 1981 and 1982. There were some changes passed to increase compliance with the tax laws. Some of the Act's provisions take effect for the 1984 return. Also, some of the tax revisions that were enacted in 1981, 1982, and 1983 first become effective in 1984. All-in-all there are a lot of 1984 changes that I will concentrate on in this article.

I and REMark need leadtime to write and publish this article so I am writing this in December, 1984. The tax forms, schedules, and tables I am using ARE NOT the final official IRS forms! There may be some small revisions made before the official IRS forms are mailed to the taxpayers sometime in January, 1985. I am using information that is released to tax consultants and publishers that reflects the current law with the latest available information. The reader should check for line number changes, rate corrections, etc. BUT, your final result should be usable when you transfer the entries from your program to the official forms by reading each line!

Both the author and REMark publish this tutorial article with the understanding that they ARE NOT engaged in rendering legal, accounting, and/or professional tax service! Please obtain professional counsel on your legal and/or accounting tax problems!

REVIEW

I hope that the readers have completed the 1983 Federal Income Tax Project discussed in "SPREADSHEET Corner-Parts 6 & 7". After you completed the project, did you "plug" your 1983 tax data into the template? Did you get the same results that you filed? If not, did you find out why?

Once in a while I will leave a "challenge" in the articles for the readers to think about. I did this in the last two articles. Did you find a substantial question? LOTUS 1-2-3 calls this problem "CIRC"--circular reference! I think that I should discuss this subject in detail!

One of the primary functions of all spreadsheet programs is to recalculate ALL the cells in a worksheet when a value or a formula in one of the cells changes. This is called "recalculation"! With the evolution of spreadsheet software, recalculation has become more important. Most first-generation spreadsheet programs gave a choice of column-wise or row-wise recalculation (1-2-3 still has it.). With column-wise recalculation the program starts the recalculating at the entry of the top-left cell, lets call it A1, and proceed down column A, then to cell B1 and down column B, etc. The row-wise recalculation starts at cell A1 and proceeds across row 1, back to cell A2 and across row 2, etc. The problem with these recalculations is that they can lead to a WRONG result, unless the template creator was very careful how the worksheet was set up. Two types of errors are found--"forward" and "circular" references. I will explain these terms with examples.

Forward reference occurs when a cell refers to another cell that is lower (or to the right of) in the worksheet. Here is an example:

```
A1 = F4
F2 = 10
F3 = 20
F4 = F2 + F3
```

As you can see, F4 should equal 30 as should A1! Now, if I change F3 to 10 and step through the recalculating process our first-generation spreadsheet program would begin recalculating at cell A1. It would still be 30 but now as we proceed either by column or row to cell F4, because I changed F3 to 10, the value of cell F4 would change to 20. How can cell A1 be equal to 30 and cell F4 equal 20? We defined A1 = F4! Thus, we have an error! The program would have to recalculate the worksheet! This is what the manuals tell you to do!

This does not remove the basic problem. I am sure that you can see that with large, complicated models it would be nearly impossible to eliminate forward references. In most cases, one extra recalculation will solve the error. LOTUS 1-2-3 and others have provided this recalculation.

When the worksheet contains circular references, one extra recalculation will not solve even a simple problem! Here is a word example of a circular reference:

```
1—Borrowings = Assets - (Total Liabilities + Equity).
2—Equity is a function of Net Income & Dividends.
3—Net Income is a function of Gross Margin & Interest Income.
4—Interest Expense & Gross Margin are a function of Borrowings.
```

Can you see this circular pattern in a everyday problem? If you do 1 or 2 recalculations, the result will still be in error!

Iterative recalculation will overcome most of the circular reference errors because with each recalculating pass through the worksheet the actual values of the problem cells will approach more closely their correct values. (ITERATIVE means a computational procedure in which replication of a cycle of operations produces results which approximate the desired result more and more closely.) Now an example:

```
A2 = .06 * A4
A3 = 100
A4 = A2 + A3
```

When these formulas are entered to the worksheet, A2 will have a value of 0, A3 will equal 100 and A4 will also equal 100. We have a circular reference! I will go through 6 iterations (recalculations) and show a table with the results after each iteration:

A2	A3	A4
1—6	100	106
2—6.36	100	106.36
3—6.3816	100	106.3816
4—6.382896	100	106.382896
5—6.38297376	100	106.38297376
6—6.382978426	100	106.382978426

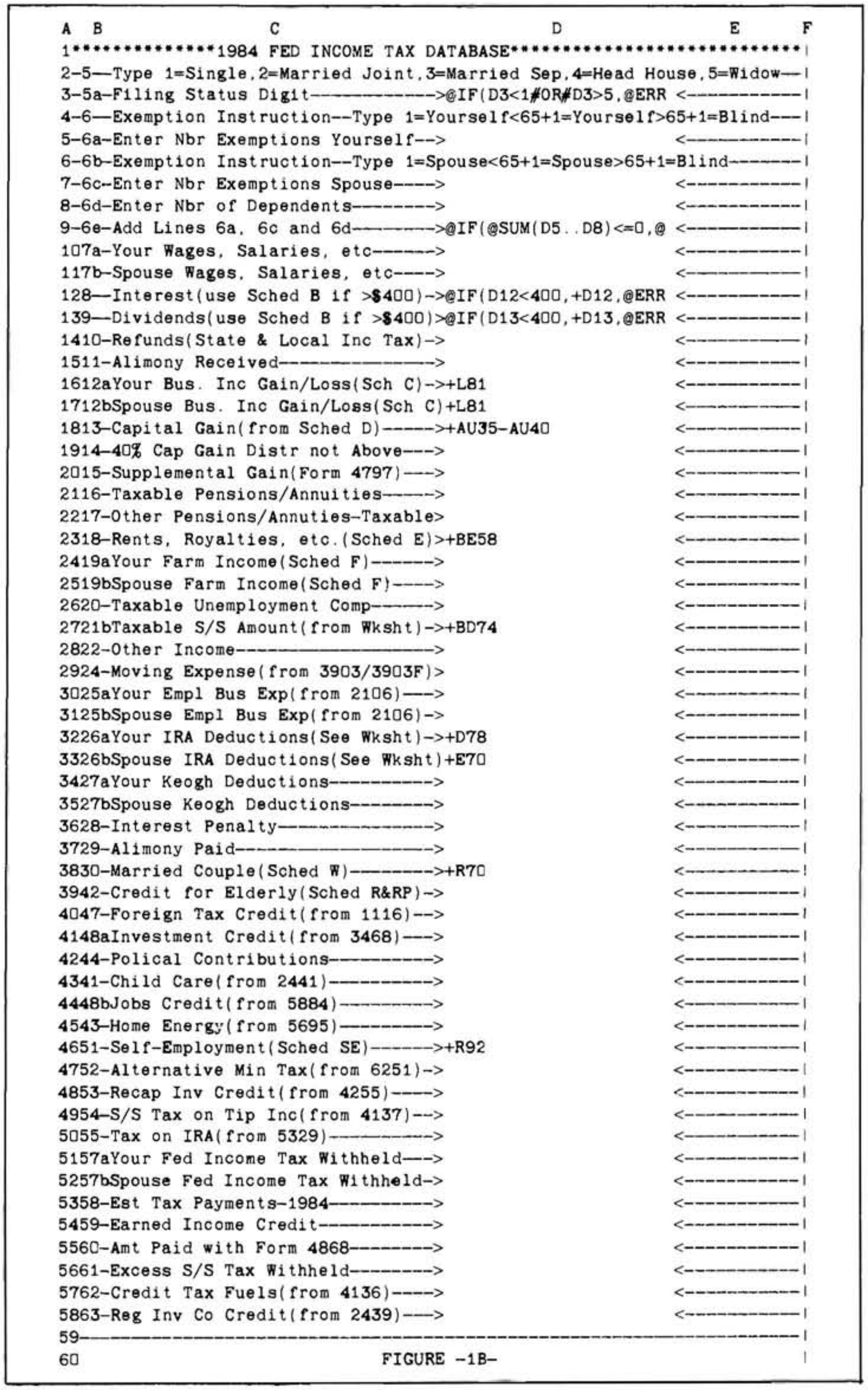
Please notice that with each iteration, the difference between the prior and the current value of cells become smaller and after 6 iterations the difference is so small as to be insignificant for most purposes. I should point out that it is possible to create worksheets with circular references that are just too complicated so that iterative recalculation is impractical because of the time that could be required for the program to do the require number of iterations! The answer is worksheet PLANNING! Avoid circular references as much as possible.

The newer generation spreadsheet programs, such as LOTUS 1-2-3, have worked on these problems. I will discuss the 1-2-3 method. LOTUS recalculates in what they call a "natural order"! This means that all the active cells in a worksheet are interrelated, and 1-2-3 does not recalculate any given cell until the cells that it depends on has been recalculated first. The relationship between the cells is seldom linear, therefore, the method of recalculation is not linear. So, the recalculation occurs in an "upside down tree" fashion, starting on the lowest level and working up. Here is an example:

```
      F5
   E3 E5 E6
D4 D5 D6 D7 D8
```

With "Natural" recalculation, the order of recalculation and the problem of forward reference is laid to rest! In the above example, cell F5 -- the most fundamental cell -- would be calculated before cell D4. This type of calculation eliminates the forward reference problem.

"Natural" recalculation is not the answer for circular references because each cell depends, directly or indirectly, on all the other cell values in the circular set. 1-2-3 would not know which cell to start with! LOTUS 1-2-3 uses a iterative recalculation that overcomes most circular references. The function key--K9--CALC--(Z-100 Version) starts the Iterative Mode. Each time you press



```
A  B                    C                           D                     E     F
1*************1984 FED INCOME TAX DATABASE****************************|
2-5—Type 1=Single,2=Married Joint,3=Married Sep,4=Head House,5=Widow—|
3-5a-Filing Status Digit------------->@IF(D3<1#OR#D3>5,@ERR <-----------|
4-6—Exemption Instruction--Type 1=Yourself<65+1=Yourself>65+1=Blind---|
5-6a-Enter Nbr Exemptions Yourself-->                       <-----------|
6-6b-Exemption Instruction--Type 1=Spouse<65+1=Spouse>65+1=Blind-------|
7-6c-Enter Nbr Exemptions Spouse---->                       <-----------|
8-6d-Enter Nbr of Dependents-------->                       <-----------|
9-6e-Add Lines 6a, 6c and 6d--------->@IF(@SUM(D5..D8)<=0,@ <-----------|
107a-Your Wages, Salaries, etc------>                       <-----------|
117b-Spouse Wages, Salaries, etc---->                       <-----------|
128—Interest(use Sched B if >$400)->@IF(D12<400,+D12,@ERR <-----------|
139—Dividends(use Sched B if >$400)>@IF(D13<400,+D13,@ERR <-----------|
1410-Refunds(State & Local Inc Tax)->                       <-----------|
1511-Alimony Received--------------->                       <-----------|
1612aYour Bus. Inc Gain/Loss(Sch C)->+L81                   <-----------|
1712bSpouse Bus. Inc Gain/Loss(Sch C)+L81                   <-----------|
1813-Capital Gain(from Sched D)----->+AU35-AU40             <-----------|
1914-40% Cap Gain Distr not Above--->                       <-----------|
2015-Supplemental Gain(Form 4797)--->                       <-----------|
2116-Taxable Pensions/Annuities----->                       <-----------|
2217-Other Pensions/Annuties-Taxable>                       <-----------|
2318-Rents, Royalties, etc.(Sched E)>+BE58                  <-----------|
2419aYour Farm Income(Sched F)------>                       <-----------|
2519bSpouse Farm Income(Sched F)---->                       <-----------|
2620-Taxable Unemployment Comp------>                       <-----------|
2721bTaxable S/S Amount(from Wksht)->+BD74                  <-----------|
2822-Other Income------------------->                       <-----------|
2924-Moving Expense(from 3903/3903F)>                       <-----------|
3025aYour Empl Bus Exp(from 2106)--->                       <-----------|
3125bSpouse Empl Bus Exp(from 2106)->                       <-----------|
3226aYour IRA Deductions(See Wksht)->+D78                   <-----------|
3326bSpouse IRA Deductions(See Wksht)+E70                   <-----------|
3427aYour Keogh Deductions---------->                       <-----------|
3527bSpouse Keogh Deductions-------->                       <-----------|
3628-Interest Penalty--------------->                       <-----------|
3729-Alimony Paid------------------->                       <-----------|
3830-Married Couple(Sched W)-------->+R70                   <-----------|
3942-Credit for Elderly(Sched R&RP)->                       <-----------|
4047-Foreign Tax Credit(from 1116)-->                       <-----------|
4148aInvestment Credit(from 3468)--->                       <-----------|
4244-Polical Contributions---------->                       <-----------|
4341-Child Care(from 2441)---------->                       <-----------|
4448bJobs Credit(from 5884)--------->                       <-----------|
4543-Home Energy(from 5695)--------->                       <-----------|
4651-Self-Employment(Sched SE)------>+R92                   <-----------|
4752-Alternative Min Tax(from 6251)->                       <-----------|
4853-Recap Inv Credit(from 4255)---->                       <-----------|
4954-S/S Tax on Tip Inc(from 4137)-->                       <-----------|
5055-Tax on IRA(from 5329)---------->                       <-----------|
5157aYour Fed Income Tax Withheld--->                       <-----------|
5257bSpouse Fed Income Tax Withheld->                       <-----------|
5358-Est Tax Payments-1984---------->                       <-----------|
5459-Earned Income Credit----------->                       <-----------|
5560-Amt Paid with Form 4868-------->                       <-----------|
5661-Excess S/S Tax Withheld-------->                       <-----------|
5762-Credit Tax Fuels(from 4136)---->                       <-----------|
5863-Reg Inv Co Credit(from 2439)--->                       <-----------|
59---------------------------------------------------------------------|
60                              FIGURE -1B-                              |
```

the F9 key, the worksheet will recalculate a specified number of times. The default number of iterations is 20 times which will solve most problems. LOTUS 1-2-3 uses Assembly Language and it is very fast. So, even if the problem would require 40 iterations, 1-2-3 has a practical, speedy solution.

I would suggest that you try my two examples--forward and circular references--with your spreadsheet program, if you are not using 1-2-3, and find out how it handles the problems. ALSO, remember that a little thought about the worksheet layout will eliminate many of the problems! My Income Tax Model grew without this planning so that I could present the project to you in a way that I thought would help you understand the project! You CAN and SHOULD plan your project! This is not a COPY assignment.

All of this project has been prepared using LOTUS 1-2-3 with an H/Z-100 computer. You will prepare your own "SPREADSHEET Preparation Forms" and revise them to work with your spreadsheet program and computer set-up using mine for reference. Start with your 1983 project; BUT, when you SAVE, use a NEW FILENAME! Be sure to keep your 1983 model intact. This should be a rather easy assignment. I will not use new functions, commands, or formulas. I will revise them as required. I will go through each of the forms and point out the main differences. If the complete formula does not show up, I will explain it.

1984 SPREADSHEET Preparation Forms

As before, I use and I want you to use the "SPREADSHEET Preparation Form" as the basis for the 1984 update project. I cannot repeat this too often. This is the only way to go. Remember to plan your worksheet using the forms that you will require for your tax return. If I have not supplied the one you need, you should be able to add it using the same methods that I have used. If you need to save working memory (RAM), you might consider eliminating some of the blank lines, separator lines, etc. I am including the following revised (labeled with a "B") forms:

```
1—1984 DATABASE-------------FIGURE -1B-
2—FORM1040-PAGE1------------FIGURE -4B-
3—FORM1040-PAGE2------------FIGURE -5B-
4—SCHEDULE B----------------FIGURE -2B-
5--SCHEDULE A---------------FIGURE -3B-
6—SCHEDULES X, Y, & Z-------FIGURE -6B-
7—IRAWORKSHEET--------------FIGURE -7B-
8--SCHEDULE C---------------FIGURE -8B-
9—SCHEDULE W----------------FIGURE -9B-
10 SCHEDULE SE--------------FIGURE -10B-
11-SCHEDULE D---------------FIGURE -11B-
12-SCHEDULE E---------------FIGURE -12B-
13-NON-ITEMIZED WORKSHEET---FIGURE -13B-
14-TAXABLE S/S WORKSHEET----FIGURE -14B-
```

Select the ones that you will need and go to work! You might want to do them all just for the learning experience if you have enough RAM.

1984 DATABASE

Figure -1B- shows the revised 1984 Federal Income Tax Database SPREADSHEET Preparation Form. The name heading has been changed to add 1984 and many of the line numbers had to be changed. Beginning in 1984, the disability income exclusion is repealed and replaced with a credit for the permanently and totally disabled, so line 30 Form 1040(83) has been removed. Beginning in 1984, a portion of Social Security or tier 1 Railroad Retirement Benefits may be includable in the gross income. Line 21b has been added for this entry and a Temporary Non-Official Worksheet has been added as Figure -14B- for the necessary calculations.

Line 54 Form 1040(83) has been eliminated, uncollected employee social security tax and RRTA tax on tips (Form W-2). This item, if any, should be included on line 56 of Form 1040(84). Also, note the entries in column D from other forms have changed and change 1983 to 1984 on line 58--Est Tax Payments--1984.

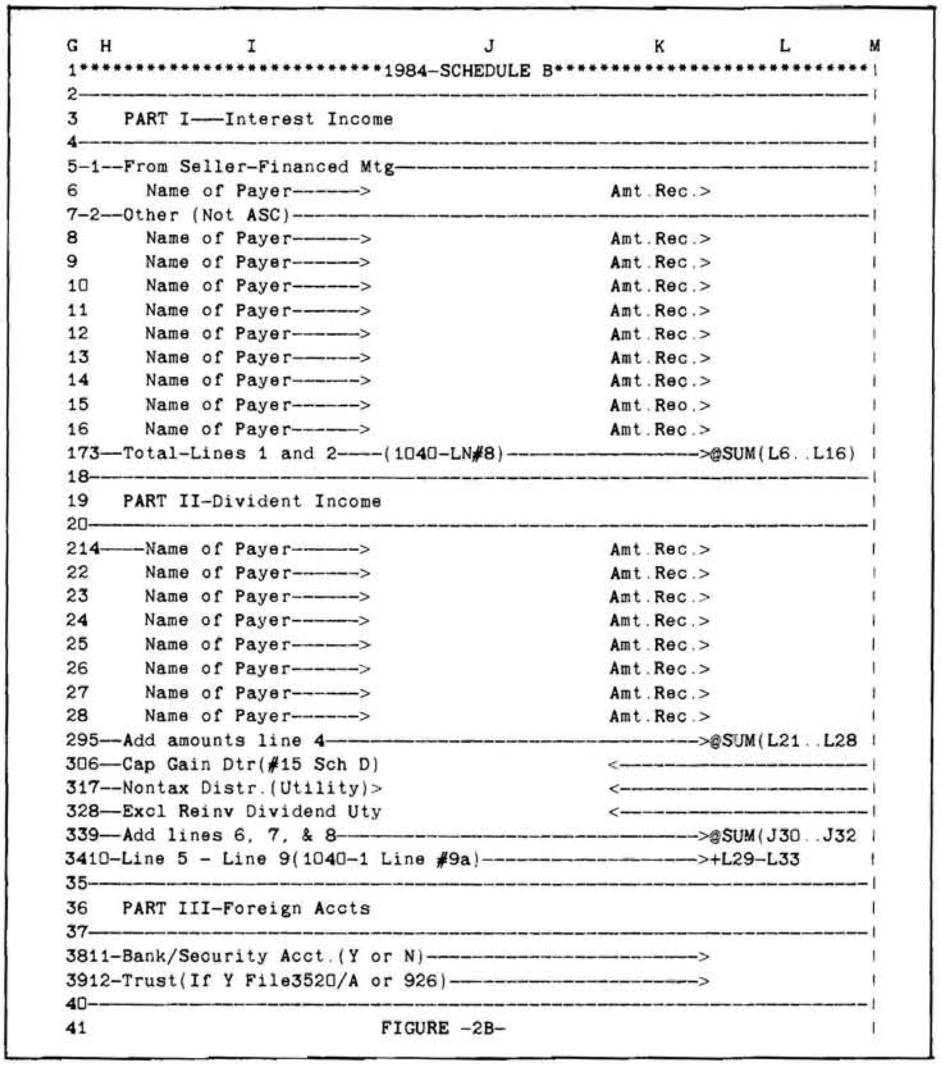
Thus, all of the changes are rather simple to make.

SCHEDULE B FORM 1984

The big change is the elimination of the section--Interest from All-Savers Certificates(ASCs). This changes many line number and formula references. The formula data shows on Figure -2B-. Don't forget to change the date to 1984!

SCHEDULE A FORM 1984

The change that really effects most tax payers is that Medicines and Drugs are no longer separated and calculated at 1% of line 33, Form1040. Beginning in 1984, all medical expenses are totaled together. Then 5% of line 33 Form 1040(84) is subtracted with the balance, if over 0, becoming the medical deduction. Only one of the formulas can not be viewed on Figure -3B-, it is for line 5--@IF((R12-R1133)>0,R12-R13,0).

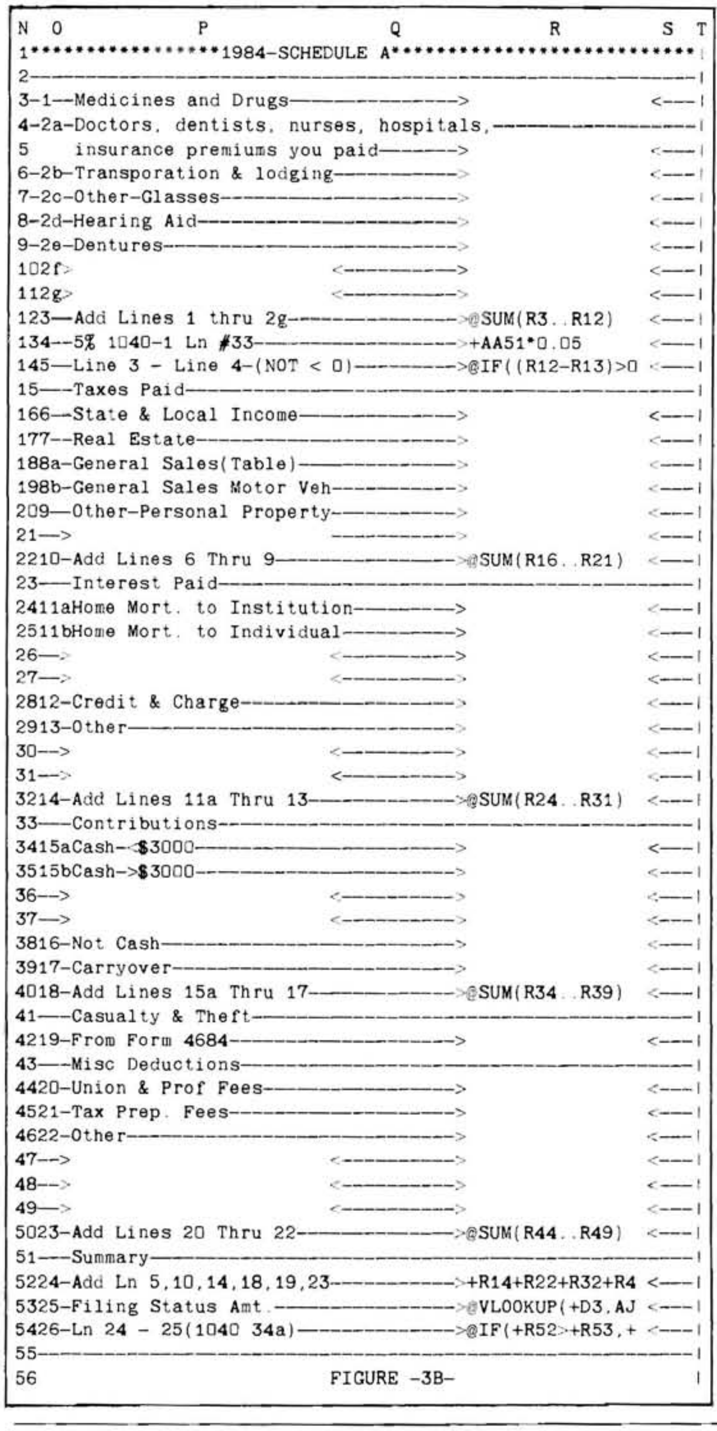


```
G H                 I                        J                 K            L        M
1*************************1984-SCHEDULE B*************************|
2-----------------------------------------------------------------|
3    PART I---Interest Income                                     |
4-----------------------------------------------------------------|
5-1--From Seller-Financed Mtg-------------------------------------|
6      Name of Payer------>                    Amt.Rec.>           |
7-2--Other (Not ASC)----------------------------------------------|
8      Name of Payer------>                    Amt.Rec.>           |
9      Name of Payer------>                    Amt.Rec.>           |
10     Name of Payer------>                    Amt.Rec.>           |
11     Name of Payer------>                    Amt.Rec.>           |
12     Name of Payer------>                    Amt.Rec.>           |
13     Name of Payer------>                    Amt.Rec.>           |
14     Name of Payer------>                    Amt.Rec.>           |
15     Name of Payer------>                    Amt.Reo.>           |
16     Name of Payer------>                    Amt.Rec.>           |
173--Total-Lines 1 and 2----(1040-LN#8)---------------->@SUM(L6..L16) |
18----------------------------------------------------------------|
19   PART II-Divident Income                                      |
20----------------------------------------------------------------|
214----Name of Payer------>                    Amt.Rec.>           |
22     Name of Payer------>                    Amt.Rec.>           |
23     Name of Payer------>                    Amt.Rec.>           |
24     Name of Payer------>                    Amt.Rec.>           |
25     Name of Payer------>                    Amt.Rec.>           |
26     Name of Payer------>                    Amt.Rec.>           |
27     Name of Payer------>                    Amt.Rec.>           |
28     Name of Payer------>                    Amt.Rec.>           |
295--Add amounts line 4-------------------------------->@SUM(L21..L28 |
306--Cap Gain Dtr(#15 Sch D)                   <-------------------|
317--Nontax Distr.(Utility)>                   <-------------------|
328--Excl Reinv Dividend Uty                   <-------------------|
339--Add lines 6, 7, & 8------------------------------->@SUM(J30..J32 |
3410-Line 5 - Line 9(1040-1 Line #9a)----------------->+L29-L33     |
35----------------------------------------------------------------|
36   PART III-Foreign Accts                                       |
37----------------------------------------------------------------|
3811-Bank/Security Acct.(Y or N)---------------------->            |
3912-Trust(If Y File3520/A or 926)-------------------->            |
40----------------------------------------------------------------|
41                         FIGURE -2B-                            |
```

For 1984, you may deduct up to $50 per night for lodging while away from home for recommended outpatient care, and medical transportation can be calculated at 9 cents per mile plus tolls and parking if you use your car. Otherwise, use your actual out of pocket expenses for other modes of transportation.

The rest of the form's expenses are figured the same way as for 1983. The formulas down in the summary section change slightly because of the new line numbers.

```
#24—+R14+R22+R32+R40+R42+R50
#25—@VLOOKUP(+D3,AJ2..AK6,1)
#26—@IF(+R52>+R53,+R52-R53,@NA)
```



```
N  O              P                 Q                R          S  T
1******************1984-SCHEDULE A*******************************|
2----------------------------------------------------------------|
3-1--Medicines and Drugs------------->                      <---|
4-2a-Doctors, dentists, nurses, hospitals,----------------------|
5    insurance premiums you paid------->                     <---|
6-2b-Transporation & lodging---------->                      <---|
7-2c-Other-Glasses-------------------->                      <---|
8-2d-Hearing Aid---------------------->                      <---|
9-2e-Dentures------------------------->                      <---|
102f>                     <---------->                      <---|
112g>                     <---------->                      <---|
123—Add Lines 1 thru 2g-------------->@SUM(R3..R12)         <---|
134--5% 1040-1 Ln #33----------------->+AA51*0.05           <---|
145—Line 3 - Line 4-(NOT < 0)--------->@IF((R12-R13)>0      <---|
15---Taxes Paid--------------------------------------------------|
166—State & Local Income------------->                      <---|
177--Real Estate---------------------->                      <---|
188a-General Sales(Table)------------>                      <---|
198b-General Sales Motor Veh--------->                      <---|
209—Other-Personal Property---------->                      <---|
21—>                      ----------->                      <---|
2210-Add Lines 6 Thru 9-------------->@SUM(R16..R21)        <---|
23---Interest Paid-----------------------------------------------|
2411aHome Mort. to Institution------->                      <---|
2511bHome Mort. to Individual-------->                      <---|
26—>                      <---------->                      <---|
27—>                      <---------->                      <---|
2812-Credit & Charge----------------->                      <---|
2913-Other--------------------------->                      <---|
30-->                     <---------->                      <---|
31-->                     <---------->                      <---|
3214-Add Lines 11a Thru 13----------->@SUM(R24..R31)        <---|
33---Contributions-----------------------------------------------|
3415aCash-<$3000--------------------->                      <---|
3515bCash->$3000--------------------->                      <---|
36-->                     <---------->                      <---|
37—>                      <---------->                      <---|
3816-Not Cash------------------------>                      <---|
3917-Carryover----------------------->                      <---|
4018-Add Lines 15a Thru 17----------->@SUM(R34..R39)        <---|
41---Casualty & Theft--------------------------------------------|
4219-From Form 4684------------------>                      <---|
43---Misc Deductions---------------------------------------------|
4420-Union & Prof Fees--------------->                      <---|
4521-Tax Prep. Fees------------------>                      <---|
4622-Other--------------------------->                      <---|
47-->                     <---------->                      <---|
48-->                     <---------->                      <---|
49—>                      <---------->                      <---|
5023-Add Lines 20 Thru 22------------>@SUM(R44..R49)        <---|
51---Summary-----------------------------------------------------|
5224-Add Ln 5,10,14,18,19,23--------->+R14+R22+R32+R4       <---|
5325-Filing Status Amt.-------------->@VLOOKUP(+D3,AJ       <---|
5426-Ln 24 - 25(1040 34a)------------>@IF(+R52>+R53,+       <---|
55---------------------------------------------------------------|
56                        FIGURE -3B-                           |
```

FORM1040-PAGE1 1984

Figure -4B- shows the changes for 1984, such as the date in the header. Some of the data item references have changed, so go over each line. Lines 21a & 21b are added to include the new Social Security tax. This will change many of the line numbers. Line 30 Form1040(83) has been eliminated because the new regulations have changed the way Disability Income Exclusion is handled. The 1984 date on line 25b Form1040(83) should be changed to 1985.

The following formulas have changed and do not show on the form:

```
LINE #32—@SUM(AA12..AA30)
LINE #43—@SUM(AA35..AA42)
LINE #44—+AA32-AA43
```

FORM1040-PAGE2 1984

Figure -5B- shows the 1984 changes. Again, watch for line number and changed data references. These changes cause some formulas to change. The following formulas do not show completely in the figure:

```
LINE #53—+AA50-AA51-AA52
LINE #55—+AA53-AA54
LINE #66—@SUM(Y62..Y65)
LINE #67—+AA59-AA66
LINE #71—@IF(AA67>AA70,AA67-AA70,0)
LINE #78—@SUM(AA71..AA77)
LINE #89—@SUM(Y81..Y87)
LINE #91—@IF(AA89>AA78,AA89-AA78,0)
LINE #94—@IF(AA78>AA89,AA78-AA89,0)
```

SCHEDULES X, Y, & Z (1984)

Figure -6B- is nearly all new because the tax rates were changed. I will provide the revised formulas below:

```
LINE #2—@IF(D3=5,2000000,1000000*D3)
LINE #4—@VLOOKUP(AF3+AF3,AD10..AH73,1)
LINE #5—+AF3-@VLOOKUP(AF2+AF3,AD10..AH73,4)
LINE #6—+AF5*@VLOOKUP(AF2+AF3,AD10..AH73,3)
```

IRAWORKSHEET 1984

Change the dates in lines 63 & 64 Figure -7B- to 1984 & 1985. The only formulas that do not show are as follows:

```
LINE #66—+D11+D17+D25-D35
LINE #70—@MIN(E65,E68,E69)
```

SCHEDULE C 1984

Figure -8B- shows all the changes required and it is a simple job of going down each line.

SCHEDULE W 1984

Figure -9B- shows the changed date to 1984 in the header. Line 65 has a change in the number of items used. I will show the changed forumlas below:

```
LINE #65—+D30+D32+D34 and +D31+D33+D35
LINE #68—@MIN(Q66,R66,30000)
```

SCHEDULE SE 1984

Figure -10B- shows some revised data references and formulas. The Max. Earnings has been raised to 37,800 and the tax rate has been raised to 0.113. The following formulas do not show up completely and are provided for your reference:

```
LINE #85—@IF((R83+R84)>400,R83+R84,0)
LINE #91—@MIN(R85,R90,37800)
```

SCHEDULE D 1984

Figure -11B- shows that a number of changes have been made effective this tax year. Please note that Parts IV and V have many changes and that a Part VI has been added. As of this writing I have not read of all the changes covering Schedule D, so if you are using this schedule be sure to review the tax consequences. The figure does show the new format. I will show all the formulas that are not completely exposed below:

```
LINE #11--@SUM(AT4..AT10) AND @SUM(AU4..AU10)
LINE #12—+AU11-AT11
LINE #14—-+AU12-AT12
LINE #24—@SUM(AT16..AT23) AND @SUM(AU16..AU23)
LINE #25—-+AU24-AT24
LINE #28—@SUM(AU25..AU27)
LINE #30--+AU28-AU29
LINE #32—+AU14+AU30
LINE #33—@IF(@MIN(AU30,AU32)<=0#OR#AU30=0,0,@MIN(AU30,AU32))
LINE #34—+AT33*0.6
LINE #35—-+AU32-AU34
LINE #37—@IF(AU32<0#AND#AU14>=0,AU32*0.5,0)
LINE #38—@IF(AU32<0#AND#AU30>=0,AU32,0)
LINE #39—@IF(AU32<0#AND#AU14<0#AND#AU30<0,(AU14+AU30)*0.5,0)
LINE #40—@MIN(AU41,AU42,AU43)
LINE #41—@SUM(AU37..AU39)
LINE #42—@IF(D3=3,1500,3000)
LINE #45—@IF(AU14<=0,AU14,0)
LINE #46—-@IF(AU30>0,AU30,0)
LINE #47—+AU46-AU45
LINE #48—@MIN(AU40,AU47)
LINE #49—-+AU47-AU48
LINE #50—-+AU40-AU48
LINE #51--@IF(AU30<0,AU30,0)
LINE #52--@IF(AU14>0,AU14,0)
LINE #53—-+AU52-AU51
LINE #55—-+AU53-AU54
LINE #62—-+AU60+AU61
LINE #64—-+AU62-AU63
LINE #72—@SUM(AU66..AU71)
```

SCHEDULE E 1984

Figure -12B- shows that except for changing the date to 1984 there are no changes to report.

NON-ITEMIZED DEDUCTION 1984

Figure -13B- does not have any changes this year. I will provide the formula below:

```
@IF(D3=3,@MAX(E85,12.5),@MAX(E85,25))
```

SOCIAL SECURITY WORKSHEET 1984

Figure -14B- shows a temporary worksheet that I put together so that you readers would know that you should look for one in your IRS Tax Package. As stated earlier, some tax payers that receive Social Security Benefits will be required to pay tax on a portion of their payments if their "modified adjusted gross income" exceeds the "base amount". I HAVE NOT seen the official worksheet so this is about the way I would expect it to look.

I will complete the formulas for you below:

```
LINE #2--(BD63+BD64)*0.5
LINE #7--@SUM(BC65..BC69)
LINE #8—-+BD71*0.5
LINE #9—@IF(D3=2,32000,25000)
LINE #10-@IF((BC65+BC70-BC72)>0,+BC65+BC70-BC72,0)
```

REMARKS

If some of you readers would like to "professionalize" your project, there are a number of improvements that could be added quite easily. I will list a few below:

1--Create a "menu" that would help the user go to the form they would like to see or use.

2--Create an "instruction sheet" that would tell the user how to use this model.

3--Protect all cells except the ones that require entries! I think that this is an important item. I will discuss this a little below using 1-2-3.

Once you have finished the complete template and you are fairly sure that you will not want to make any changes you should protect the template! When 1-2-3 starts every cell is "protected", but the global protection command is disabled. This means that all the cells in the worksheet can be modified. So, first we must enable global protection using the following command:

```
/Worksheet Global Protection Enable and Return.|
```

Once this command is issued, all worksheet cells are protected. Now you can selectively unprotect certain cells/ranges using this command:

```
/Range Unprotect and Return.
```

This prepares these cells for data entry. Of course, you can reprotect any of these cells with /Range Protect command. I find that if you need to make a formula or label change it is best to use this command:

```
/Worksheet Global Protection Disable and Return.
```

After the changes are made, restore the protection again with the Enable command.

1-2-3 has even more protection, the movement of the cursor can be limited by using the following command:

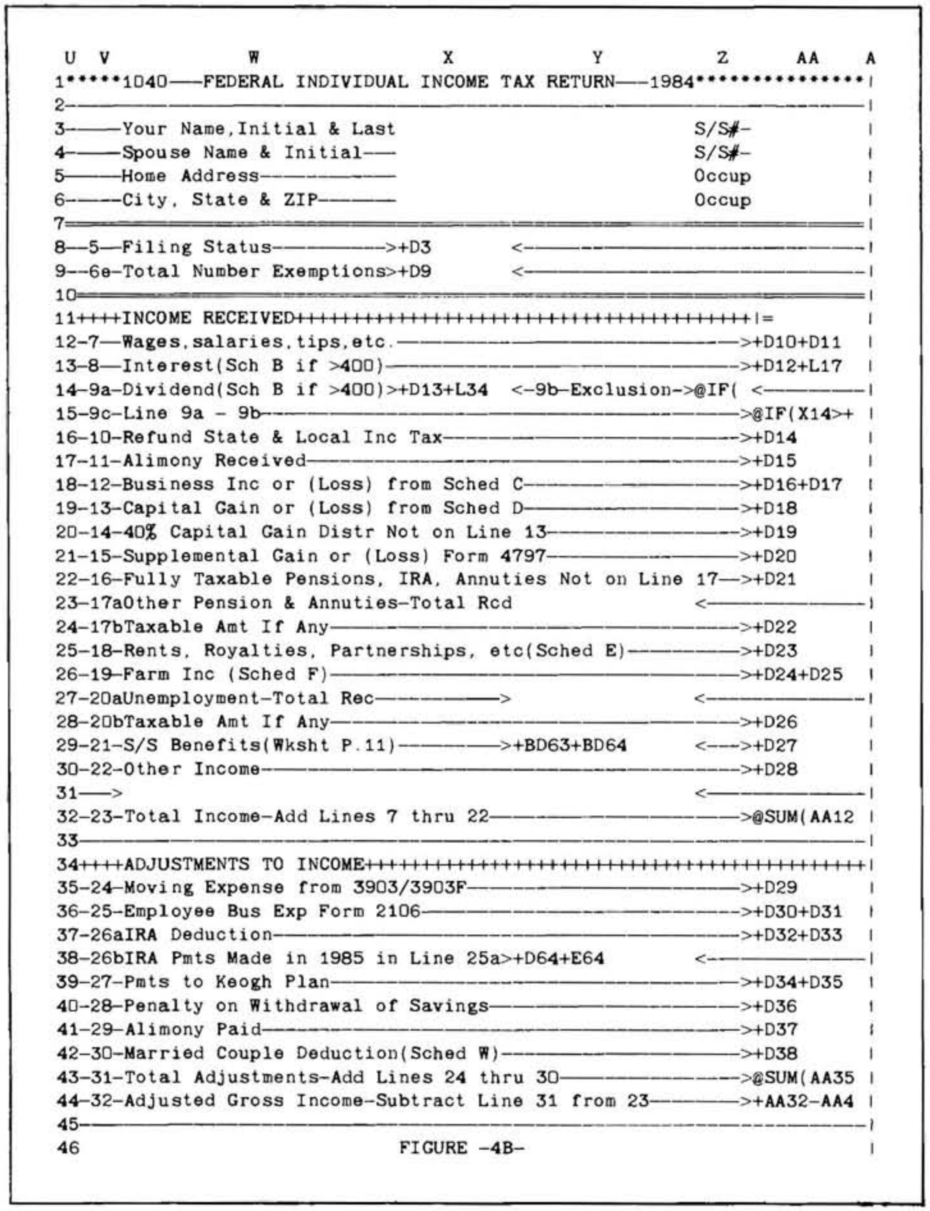
```
/Range Input and Return.
```

This command will allow the cursor to move only to /Range Unprotected cells and must be used when the special form-entry areas discussed are set up. I would suggest that you read your manual and experiment with a COPY of your template.

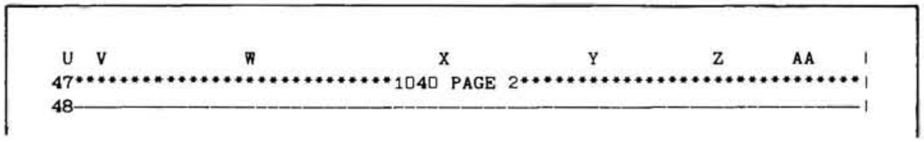
CONCLUSION

If you are following the NEWS, it sounds like there will be many changes coming for the 1985 Income Tax Return and for later years. Thus, I am sure that you can see the need for knowing how to revise the Federal Income Tax Model for each year. In fact, you may want to start making the changes during the year so that you can use the revised model for your tax planning during the year. Preparing tax returns can be pretty boring but I hope that this project will make it a little more interesting.

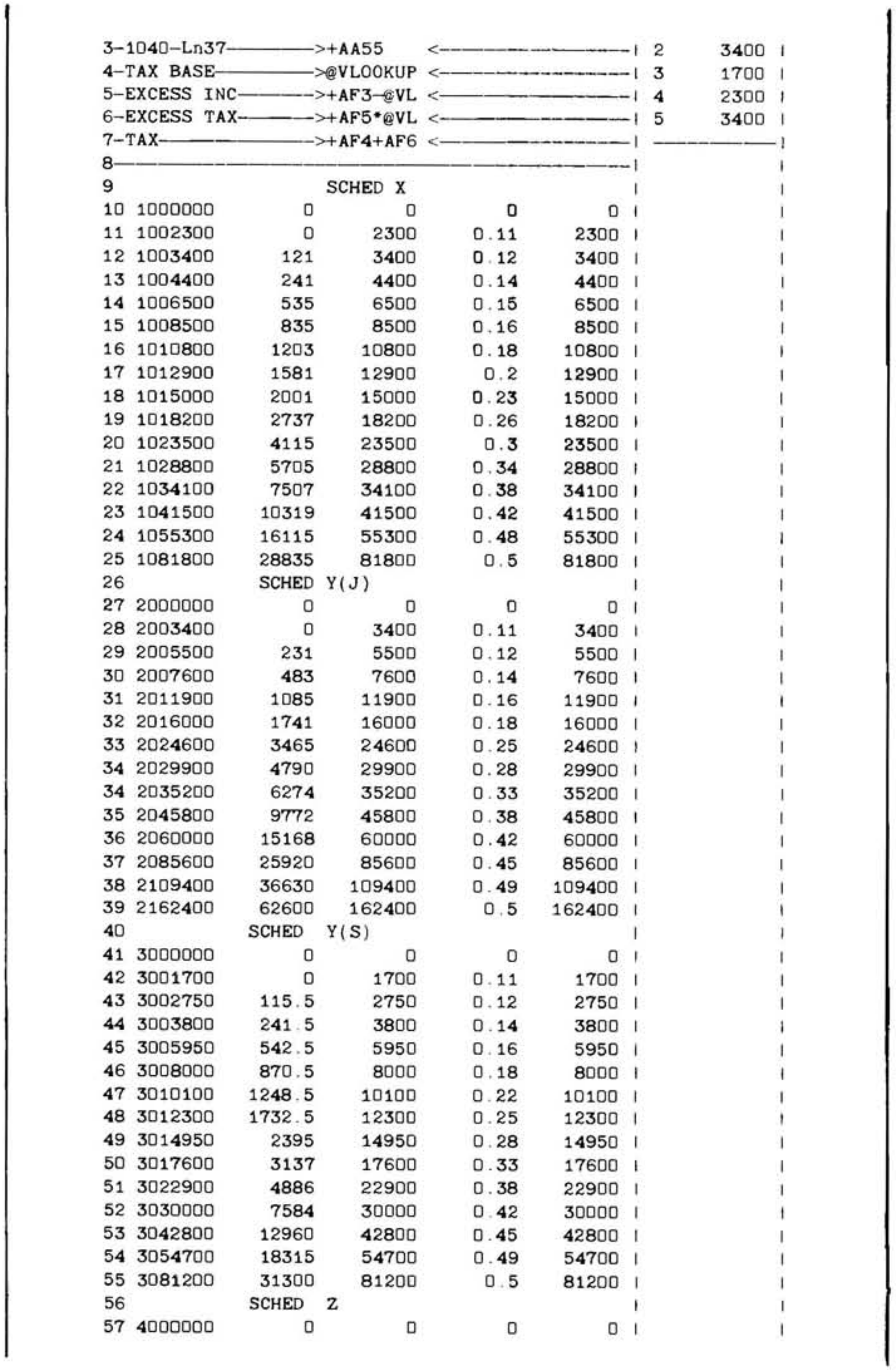
"SPREADSHEET Corner" will be showing the readers how to set you tax data up into a database in a future article. I am sure that you know that the IRS requires you to keep your records for a number of years. This would make this easy. It would also provide a place to "store" your tax data during the year so that at tax time all the data would be at your finger tips! The data is always available for tax planning as well and at tax preparation time doing your tax return will not be a bore!



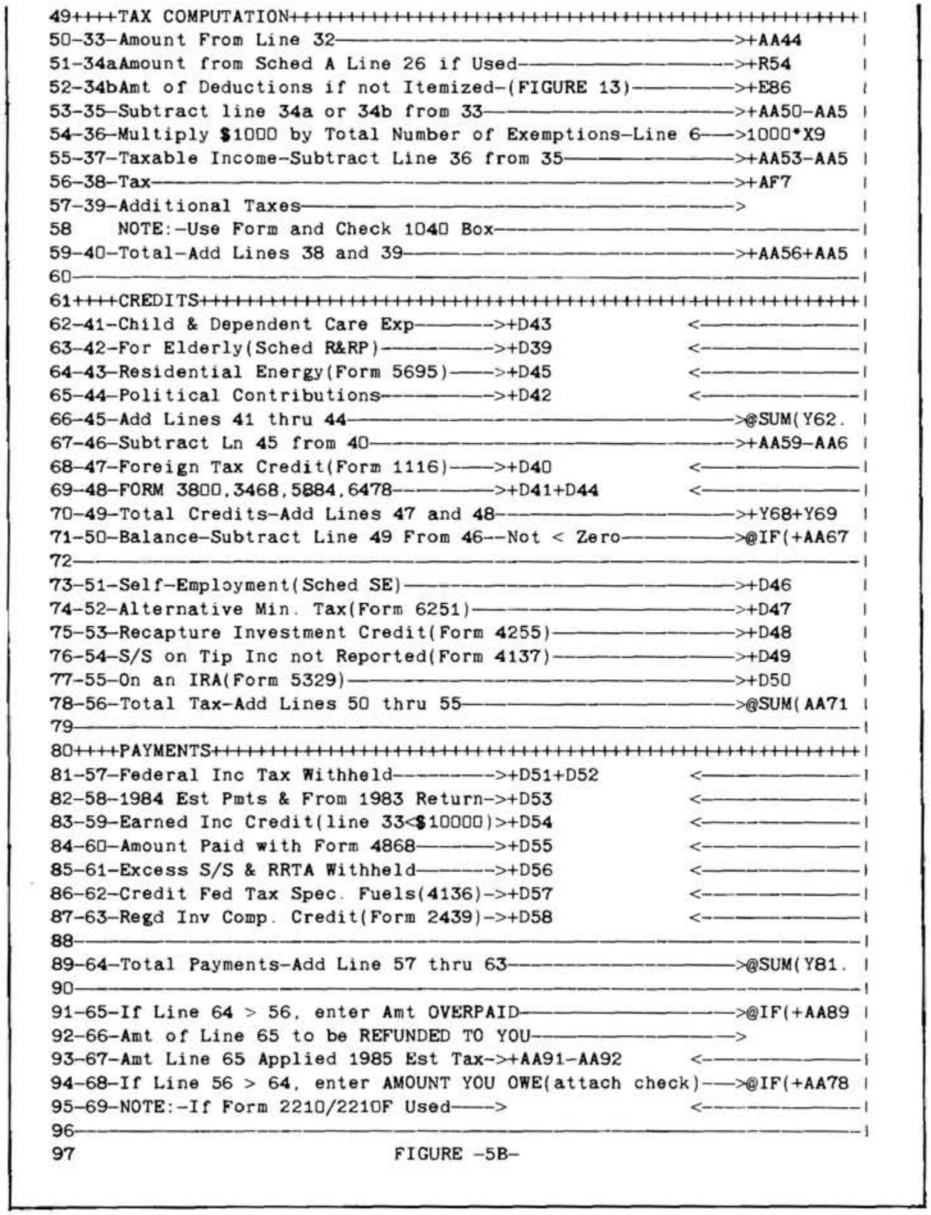
```
U V           W              X           Y           Z      AA    A
1*****1040---FEDERAL INDIVIDUAL INCOME TAX RETURN---1984**************|
2---------------------------------------------------------------------|
3-----Your Name,Initial & Last                           S/S#-        |
4-----Spouse Name & Initial---                           S/S#-        |
5-----Home Address------------                           Occup        |
6-----City, State & ZIP-------                           Occup        |
7=====================================================================|
8--5--Filing Status---------->+D3      <------------------------------|
9--6e-Total Number Exemptions>+D9      <------------------------------|
10====================================================================|
11++++INCOME RECEIVED++++++++++++++++++++++++++++++++++++++++++++|=   |
12-7--Wages,salaries,tips,etc.-------------------------->+D10+D11     |
13-8--Interest(Sch B if >400)-------------------------->+D12+L17      |
14-9a-Dividend(Sch B if >400)>+D13+L34  <-9b-Exclusion->@IF( <--------|
15-9c-Line 9a - 9b---------------------------------->@IF(X14>+        |
16-10-Refund State & Local Inc Tax-------------------->+D14           |
17-11-Alimony Received-------------------------------->+D15           |
18-12-Business Inc or (Loss) from Sched C------------->+D16+D17       |
19-13-Capital Gain or (Loss) from Sched D------------->+D18           |
20-14-40% Capital Gain Distr Not on Line 13----------->+D19           |
21-15-Supplemental Gain or (Loss) Form 4797----------->+D20           |
22-16-Fully Taxable Pensions, IRA, Annuities Not on Line 17-->+D21    |
23-17aOther Pension & Annuities-Total Rcd              <--------------|
24-17bTaxable Amt If Any------------------------------>+D22           |
25-18-Rents, Royalties, Partnerships, etc(Sched E)---->+D23           |
26-19-Farm Inc (Sched F)------------------------------>+D24+D25       |
27-20aUnemployment-Total Rec-------->                  <--------------|
28-20bTaxable Amt If Any------------------------------>+D26           |
29-21-S/S Benefits(Wksht P.11)------>+BD63+BD64        <--->+D27      |
30-22-Other Income------------------------------------>+D28           |
31--->                                                 <--------------|
32-23-Total Income-Add Lines 7 thru 22---------------->@SUM(AA12      |
33--------------------------------------------------------------------|
34++++ADJUSTMENTS TO INCOME+++++++++++++++++++++++++++++++++++++++++++|
35-24-Moving Expense from 3903/3903F------------------>+D29           |
36-25-Employee Bus Exp Form 2106---------------------->+D30+D31       |
37-26aIRA Deduction----------------------------------->+D32+D33       |
38-26bIRA Pmts Made in 1985 in Line 25a>+D64+E64       <--------------|
39-27-Pmts to Keogh Plan------------------------------>+D34+D35       |
40-28-Penalty on Withdrawal of Savings---------------->+D36           |
41-29-Alimony Paid------------------------------------>+D37           |
42-30-Married Couple Deduction(Sched W)--------------->+D38           |
43-31-Total Adjustments-Add Lines 24 thru 30---------->@SUM(AA35      |
44-32-Adjusted Gross Income-Subtract Line 31 from 23-->+AA32-AA4      |
45--------------------------------------------------------------------|
46                        FIGURE -4B-                                 |
```



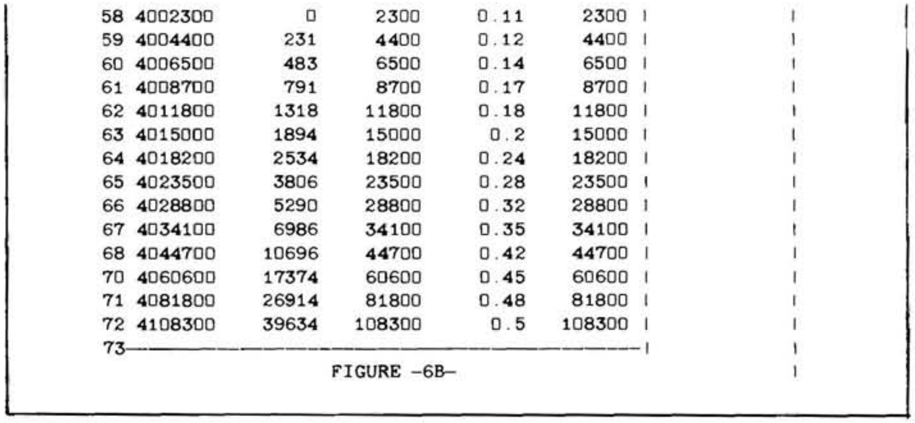
```
U V           W              X           Y           Z      AA    |
47***************************1040 PAGE 2**************************|
48----------------------------------------------------------------|
```



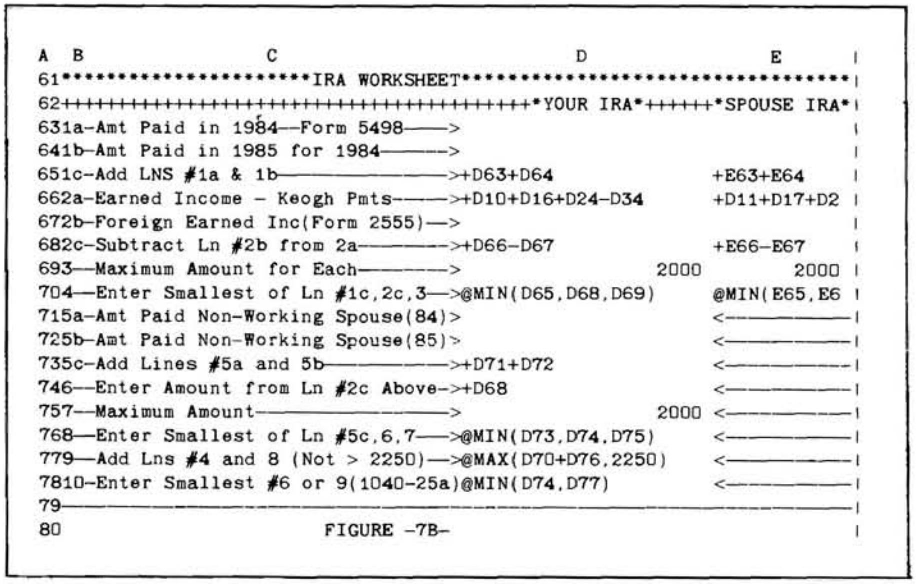
```
3-1040-Ln37--------->+AA55    <-----------------|  2      3400 |
4-TAX BASE---------->@VLOOKUP <-----------------|  3      1700 |
5-EXCESS INC-------->+AF3-@VL <-----------------|  4      2300 |
6-EXCESS TAX-------->+AF5*@VL <-----------------|  5      3400 |
7-TAX--------------->+AF4+AF6 <-----------------|  ------------|
8-----------------------------------------------|              |
9                 SCHED X                       |              |
10 1000000       0       0       0        0     |              |
11 1002300       0    2300    0.11     2300     |              |
12 1003400     121    3400    0.12     3400     |              |
13 1004400     241    4400    0.14     4400     |              |
14 1006500     535    6500    0.15     6500     |              |
15 1008500     835    8500    0.16     8500     |              |
16 1010800    1203   10800    0.18    10800     |              |
17 1012900    1581   12900     0.2    12900     |              |
18 1015000    2001   15000    0.23    15000     |              |
19 1018200    2737   18200    0.26    18200     |              |
20 1023500    4115   23500     0.3    23500     |              |
21 1028800    5705   28800    0.34    28800     |              |
22 1034100    7507   34100    0.38    34100     |              |
23 1041500   10319   41500    0.42    41500     |              |
24 1055300   16115   55300    0.48    55300     |              |
25 1081800   28835   81800     0.5    81800     |              |
26            SCHED Y(J)                        |              |
27 2000000       0       0       0        0     |              |
28 2003400       0    3400    0.11     3400     |              |
29 2005500     231    5500    0.12     5500     |              |
30 2007600     483    7600    0.14     7600     |              |
31 2011900    1085   11900    0.16    11900     |              |
32 2016000    1741   16000    0.18    16000     |              |
33 2024600    3465   24600    0.25    24600     |              |
34 2029900    4790   29900    0.28    29900     |              |
34 2035200    6274   35200    0.33    35200     |              |
35 2045800    9772   45800    0.38    45800     |              |
36 2060000   15168   60000    0.42    60000     |              |
37 2085600   25920   85600    0.45    85600     |              |
38 2109400   36630  109400    0.49   109400     |              |
39 2162400   62600  162400     0.5   162400     |              |
40           SCHED  Y(S)                        |              |
41 3000000       0       0       0        0     |              |
42 3001700       0    1700    0.11     1700     |              |
43 3002750   115.5    2750    0.12     2750     |              |
44 3003800   241.5    3800    0.14     3800     |              |
45 3005950   542.5    5950    0.16     5950     |              |
46 3008000   870.5    8000    0.18     8000     |              |
47 3010100  1248.5   10100    0.22    10100     |              |
48 3012300  1732.5   12300    0.25    12300     |              |
49 3014950    2395   14950    0.28    14950     |              |
50 3017600    3137   17600    0.33    17600     |              |
51 3022900    4886   22900    0.38    22900     |              |
52 3030000    7584   30000    0.42    30000     |              |
53 3042800   12960   42800    0.45    42800     |              |
54 3054700   18315   54700    0.49    54700     |              |
55 3081200   31300   81200     0.5    81200     |              |
56           SCHED  Z                           |              |
57 4000000       0       0       0        0     |              |
```



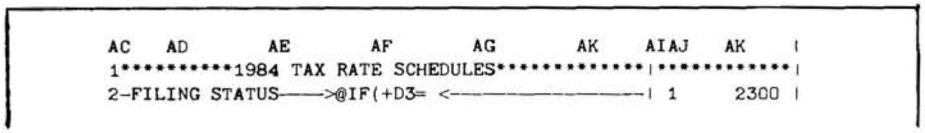
```
49++++TAX COMPUTATION++++++++++++++++++++++++++++++++++++++++++++++++++++++|
50-33-Amount From Line 32----------------------------------->+AA44       |
51-34aAmount from Sched A Line 26 if Used------------------->+R54        |
52-34bAmt of Deductions if not Itemized-(FIGURE 13)--------->+E86        |
53-35-Subtract line 34a or 34b from 33---------------------->+AA50-AA5   |
54-36-Multiply $1000 by Total Number of Exemptions-Line 6--->1000*X9     |
55-37-Taxable Income-Subtract Line 36 from 35--------------->+AA53-AA5   |
56-38-Tax--------------------------------------------------->+AF7        |
57-39-Additional Taxes-------------------------------------->            |
58    NOTE:-Use Form and Check 1040 Box--------------------------------- |
59-40-Total-Add Lines 38 and 39----------------------------->+AA56+AA5   |
60-----------------------------------------------------------------------|
61++++CREDITS++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++|
62-41-Child & Dependent Care Exp------>+D43           <------------------|
63-42-For Elderly(Sched R&RP)--------->+D39           <------------------|
64-43-Residential Energy(Form 5695)---->+D45           <------------------|
65-44-Political Contributions--------->+D42           <------------------|
66-45-Add Lines 41 thru 44---------------------------------->@SUM(Y62.   |
67-46-Subtract Ln 45 from 40-------------------------------->+AA59-AA6   |
68-47-Foreign Tax Credit(Form 1116)---->+D40           <------------------|
69-48-FORM 3800,3468,5884,6478-------->+D41+D44       <------------------|
70-49-Total Credits-Add Lines 47 and 48--------------------->+Y68+Y69    |
71-50-Balance-Subtract Line 49 From 46--Not < Zero---------->@IF(+AA67   |
72-----------------------------------------------------------------------|
73-51-Self-Employment(Sched SE)----------------------------->+D46        |
74-52-Alternative Min. Tax(Form 6251)----------------------->+D47        |
75-53-Recapture Investment Credit(Form 4255)---------------->+D48        |
76-54-S/S on Tip Inc not Reported(Form 4137)---------------->+D49        |
77-55-On an IRA(Form 5329)---------------------------------->+D50        |
78-56-Total Tax-Add Lines 50 thru 55------------------------>@SUM(AA71   |
79-----------------------------------------------------------------------|
80++++PAYMENTS+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++|
81-57-Federal Inc Tax Withheld-------->+D51+D52       <------------------|
82-58-1984 Est Pmts & From 1983 Return->+D53           <------------------|
83-59-Earned Inc Credit(line 33<$10000)>+D54           <------------------|
84-60-Amount Paid with Form 4868------>+D55           <------------------|
85-61-Excess S/S & RRTA Withheld------>+D56           <------------------|
86-62-Credit Fed Tax Spec. Fuels(4136)->+D57           <------------------|
87-63-Regd Inv Comp. Credit(Form 2439)->+D58           <------------------|
88-----------------------------------------------------------------------|
89-64-Total Payments-Add Line 57 thru 63-------------------->@SUM(Y81.   |
90-----------------------------------------------------------------------|
91-65-If Line 64 > 56, enter Amt OVERPAID------------------->@IF(+AA89   |
92-66-Amt of Line 65 to be REFUNDED TO YOU------------------>            |
93-67-Amt Line 65 Applied 1985 Est Tax->+AA91-AA92    <------------------|
94-68-If Line 56 > 64, enter AMOUNT YOU OWE(attach check)--->@IF(+AA78   |
95-69-NOTE:-If Form 2210/2210F Used---->              <------------------|
96-----------------------------------------------------------------------|
97                          FIGURE -5B-
```



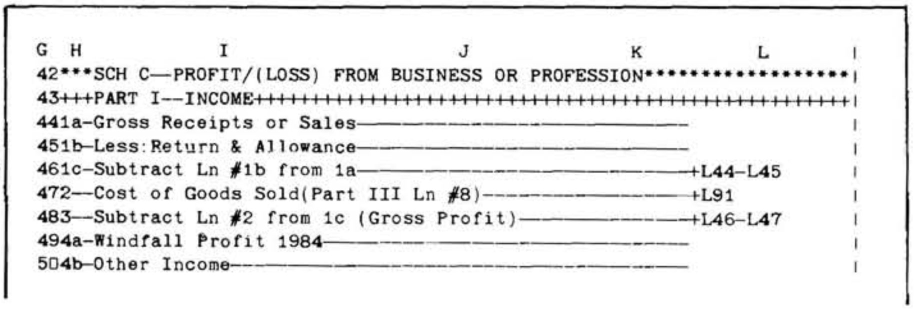
```
58 4002300        0    2300   0.11    2300 |                |
59 4004400      231    4400   0.12    4400 |                |
60 4006500      483    6500   0.14    6500 |                |
61 4008700      791    8700   0.17    8700 |                |
62 4011800     1318   11800   0.18   11800 |                |
63 4015000     1894   15000    0.2   15000 |                |
64 4018200     2534   18200   0.24   18200 |                |
65 4023500     3806   23500   0.28   23500 |                |
66 4028800     5290   28800   0.32   28800 |                |
67 4034100     6986   34100   0.35   34100 |                |
68 4044700    10696   44700   0.42   44700 |                |
70 4060600    17374   60600   0.45   60600 |                |
71 4081800    26914   81800   0.48   81800 |                |
72 4108300    39634  108300    0.5  108300 |                |
73-----------------------------------------|                |
                FIGURE -6B-                                 |
```



```
A  B                C                              D                  E   |
61*********************IRA WORKSHEET***************************************|
62++++++++++++++++++++++++++++++++++++++++*YOUR IRA*++++++*SPOUSE IRA*|
631a-Amt Paid in 1984--Form 5498---->                                     |
641b-Amt Paid in 1985 for 1984------>                                     |
651c-Add LNS #1a & 1b------------->+D63+D64                 +E63+E64      |
662a-Earned Income - Keogh Pmts---->+D10+D16+D24-D34         +D11+D17+D2   |
672b-Foreign Earned Inc(Form 2555)-->                                     |
682c-Subtract Ln #2b from 2a------->+D66-D67                 +E66-E67      |
693--Maximum Amount for Each-------->                  2000         2000  |
704--Enter Smallest of Ln #1c,2c,3-->@MIN(D65,D68,D69)      @MIN(E65,E6   |
715a-Amt Paid Non-Working Spouse(84)>                       <-------------|
725b-Amt Paid Non-Working Spouse(85)>                       <-------------|
735c-Add Lines #5a and 5b---------->+D71+D72                <-------------|
746--Enter Amount from Ln #2c Above->+D68                    <-------------|
757--Maximum Amount----------------->                  2000 <-------------|
768--Enter Smallest of Ln #5c,6,7-->@MIN(D73,D74,D75)       <-------------|
779--Add Lns #4 and 8 (Not > 2250)-->@MAX(D70+D76,2250)      <-------------|
7810-Enter Smallest #6 or 9(1040-25a)@MIN(D74,D77)          <-------------|
79------------------------------------------------------------------------|
80                        FIGURE -7B-                                     |
```



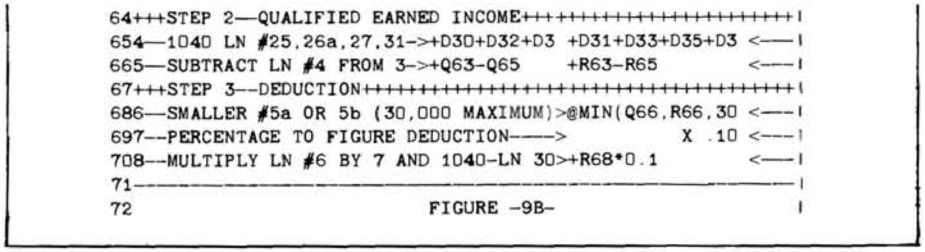
```
AC   AD      AE      AF      AG      AK    AIAJ   AK   |
1**********1984 TAX RATE SCHEDULES*************|************|
2-FILING STATUS---->@IF(+D3= <-----------------| 1     2300 |
```



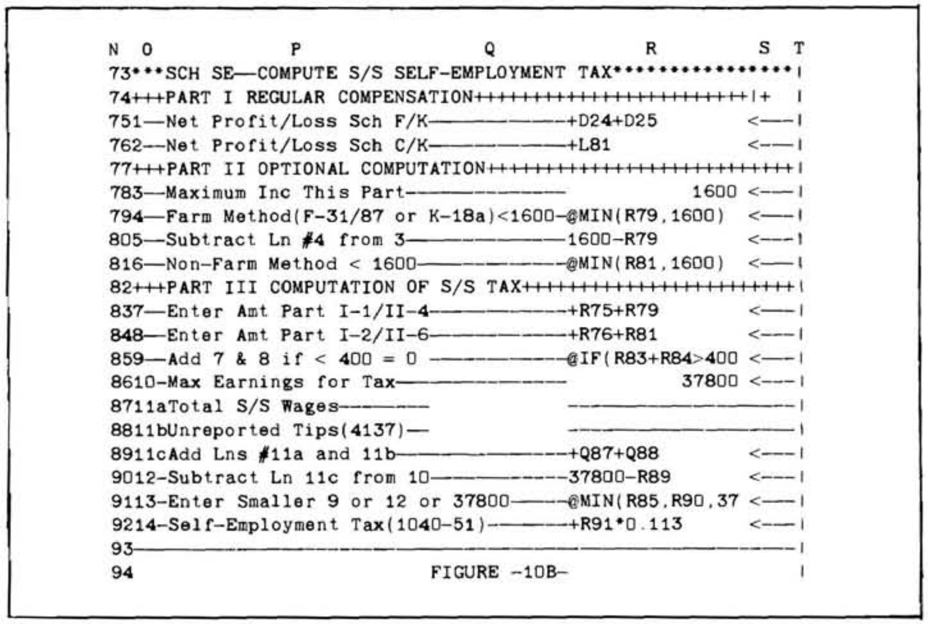
```
G  H         I               J              K          L     |
42***SCH C--PROFIT/(LOSS) FROM BUSINESS OR PROFESSION****************|
43+++PART I--INCOME++++++++++++++++++++++++++++++++++++++++++++++++++|
441a-Gross Receipts or Sales------------------------------           |
451b-Less:Return & Allowance------------------------------           |
461c-Subtract Ln #1b from 1a------------------------------+L44-L45   |
472--Cost of Goods Sold(Part III Ln #8)-------------------+L91       |
483--Subtract Ln #2 from 1c (Gross Profit)----------------+L46-L47   |
494a-Windfall Profit 1984---------------------------------           |
504b-Other Income-----------------------------------------           |
515--Add Lns #3,4a,4b-----------------------------------@SUM(L48..L50 |
52+++PART II--DEDUCTIONS++++++++++++++++++++++++++++++++++++++++++++++|
536--Advertising----------------------------------------              |
547--Bad Debts from Sales-------------------------------              |
558--Bank Service Charges-------------------------------              |
569--Car/Truck Expenses---------------------------------              |
5710-Commissions----------------------------------------              |
5811-Depletion------------------------------------------              |
5912-Depreciation(Form 4562)----------------------------              |
6013-Dues & Publications--------------------------------              |
6114-Employee Benefit Progs-----------------------------              |
6215-Freight(not Part III)------------------------------              |
6316-Insurance------------------------------------------              |
6417-Interest on Business Indebtedness------------------              |
6518-Laundry and Cleaning-------------------------------              |
6619-Legal & Prof Services------------------------------              |
6720-Office Expense-------------------------------------              |
6821-Pension and Profit-Sharing-------------------------              |
6922-Rent on Bus Property-------------------------------              |
7023-Repairs--------------------------------------------              |
7124-Supplies(not Part III)-----------------------------              |
7225-Taxes----------------------------------------------              |
7326-Travel & Entertainment ----------------------------              |
7427-Utilities & Telephone------------------------------              |
7528aWages--------------------------------------        --------------|
7628bJobs Credit--------------------------------        --------------|
7728cSubtract 28b from 28a------------------------------+K75-K76      |
7829-Windfall Profit Paid 83----------------------------              |
7930-Other Expenses-------------------------------------              |
8031-Add Lines 6 thru 30--------------------------------@SUM(L53..L79 |
8132-Subtract Ln #31 from 5 (1040-12 & Sch SE Part I-2)--+L51-L80     |
8233-If Ln #32 Loss Check---(Form 6189 Risk?)-----------  Yes     No  |
83+++PART III COST OF GOOD SOLD and/or OPERATION++++++++++++++++++++++|
841--Inventory Begin Year-------------------------------              |
852--Purchases less Personal----------------------------              |
863--Cost of Labor (not you)----------------------------              |
874--Materials & Supplies-------------------------------              |
885--Other Costs----------------------------------------              |
896--Add Lines 1 thru 5---------------------------------@SUM(L84..L88 |
907--Inventory End of Year------------------------------              |
918--COST of GOODS SOLD------(Enter Part I Ln #2)-------+L89-L90      |
92--------------------------------------------------------------------|
93                          FIGURE -8B-
```

```
  N  O            P                   Q              R            S  |
  57***SCH W--DEDUCTION FOR MARRIED COUPLE--BOTH WORK-1984****|
  58----------------------------------------------------------|
  59+++STEP 1--EARNED INCOME++++++++++++++++++++++++++++++++++|
  60---------------------------- (a) YOU      (b) SPOUSE  ----|
  611--WAGES, ETC-1040-LN #7->+D10          +D11          <---|
  622--NET PROFIT/(LOSS)-C/F/K+D16+D24      +D17+D25      <---|
  633--ADD LINES 1 & 2--------->+Q61+Q62    +R61+R62      <---|
```

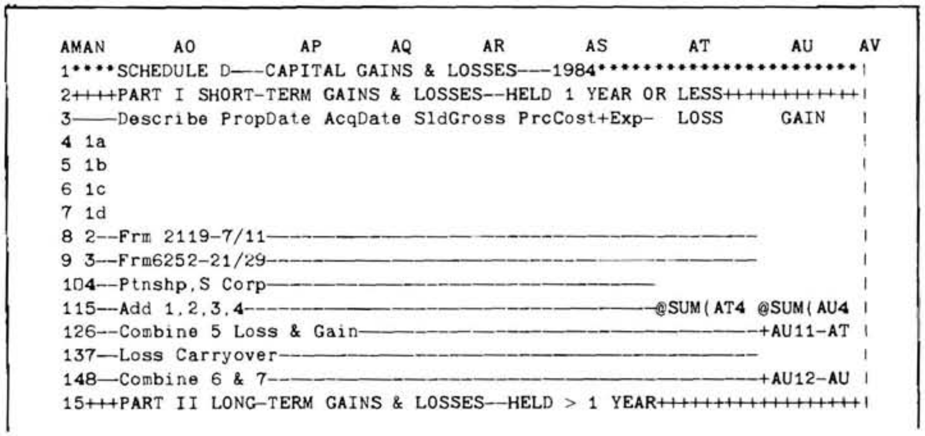
```
169a                                                                  |
179b                                                                  |
189c                                                                  |
199d                                                                  |
209e                                                                  |
2110-2119-7,11,16,18--------------------------------------------      |
2211-Frm6252-21/29----------------------------------------------      |
2312-Ptnshp,S Corp--------------------------------------              |
2413-Add9,10,11,12--------------------------------@SUM(AT1 @SUM(AU1  |
2514-Combine 13 Loss&Gain---------------------------------+AU24-AT   |
2615-Cap Gain Dist----------------------------------------------      |
2716-Gain4797-6a-1----------------------------------------------      |
2817-Combine 14-16----------------------------------------@SUM(AU2   |
2918-LossCarryover----------------------------------------------      |
3019-Combine 17-18----------------------------------------+AU28-AU    |
31+++PART III SUMMARY I-II+++++++++++++++++++++++++++++++++++++++++++|
3220-Combine8 & 19----------------------------------------+AU14+AU   |
3321-Smaller19-20,0 if loss or 19 Blank------------@IF(@MIN ---------|
3422-Enter 60% #21----------------------------------------+AT33*0.   |
3523-Subtract22 from 20(1040-13)--------------------------+AU32-AU   |
3624-If 20 < 0 do:---------------------------------------------------|
3724aIf8 => 0,Enter 50% 20--------------------------------@IF(AU32   |
3824bIf19 => 0,Enter Ln 20--------------------------------@IF(AU32   |
3924cIf8 & 19 < 0,Enter 50% 19 + 8------------------------@IF(AU32   |
4025-EnterHere & as a Loss 1040-13 smallest of:-----------@MIN(AU4   |
4125aAmt on Ln #24----------------------------------------@SUM(AU3   |
4225b$3000($1500if Married Separate Return)---------------@IF(D3=3   |
4325cAdjTaxable Income------------------------------------+AA38      |
44+++PART IV CARRYOVERS++++++++++++++++++++++++++++++++++++++++++++++|
4526-Enter8 Loss or 0,Skip 27-30, Go To 31----------------@IF(AU14   |
4627-Enter19 Gain or 0------------------------------------@IF(AU30   |
4728-Reduce26 Loss by 27 Gain-----------------------------+AU46-AU   |
4829-EnterSmallest 25/28----------------------------------@MIN(AU4   |
4930-Subtract29 from 28-----------------------------------+AU47-AU   |
5031-Subtract29 From 25-----------------------------------+AU40-AU   |
5132-Enter19 Loss or 0,Skip 33-36-------------------------@IF(AU30   |
5233-Enter8 Gain or 0-------------------------------------@IF(AU14   |
5334-Reduce32 Loss by 33 Gain-----------------------------+AU52-AU   |
5435-Multiply Ln #31 * 2----------------------------------+AU50*2    |
5536-Subtract35 From 34-----------------------------------+AU53-AU   |
56+++PART V INSTALLMENTS+++++++++++++++++++++++++++++++++++++++++++++|
57   Check Instructions----------------------------------------------|
58+++PART VI RECONCILIATION FORMS 1099-B WITH TAX REURN++++++++++++++|
59   Section A--Sales of Bonds, Stock,etc.---------------------------|
6037-Total sales of stock, bonds, etc.--------------------------      |
6138-Proceeds from Sch D not in Ln 37---------------------------      |
6239-Add 37 & 38------------------------------------------+AU60+AU   |
6340-Part 37 not reported Sch D this year-----------------------      |
6441-Subtract40 From 39-----------------------------------+AU62-AU   |
65   Section B--Bartering Income-------------------------------------|
6642-From 1040-22-----------------------------------------------      |
6743-Schedule C-------------------------------------------------      |
6844-Schedule D-------------------------------------------------      |
6945-Schedule E-------------------------------------------------      |
7046-Schedule F-------------------------------------------------      |
7147-Other(attach info sheet)-----------------------------------      |
```



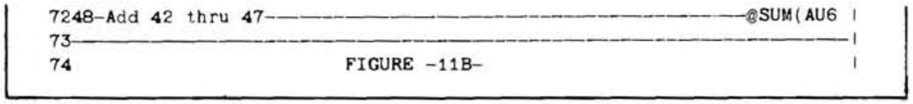
```
64+++STEP 2---QUALIFIED EARNED INCOME++++++++++++++++++++++++++++|
654---1040 LN #25,26a,27,31->+D30+D32+D3 +D31+D33+D35+D3 <---|
665---SUBTRACT LN #4 FROM 3->+Q63-Q65     +R63-R65         <---|
67+++STEP 3---DEDUCTION++++++++++++++++++++++++++++++++++++++++++|
686---SMALLER #5a OR 5b (30,000 MAXIMUM)>@MIN(Q66,R66,30 <---|
697---PERCENTAGE TO FIGURE DEDUCTION----->           X .10 <---|
708---MULTIPLY LN #6 BY 7 AND 1040-LN 30>+R68*0.1          <---|
71-----------------------------------------------------------------|
72                          FIGURE -9B-                          |
```



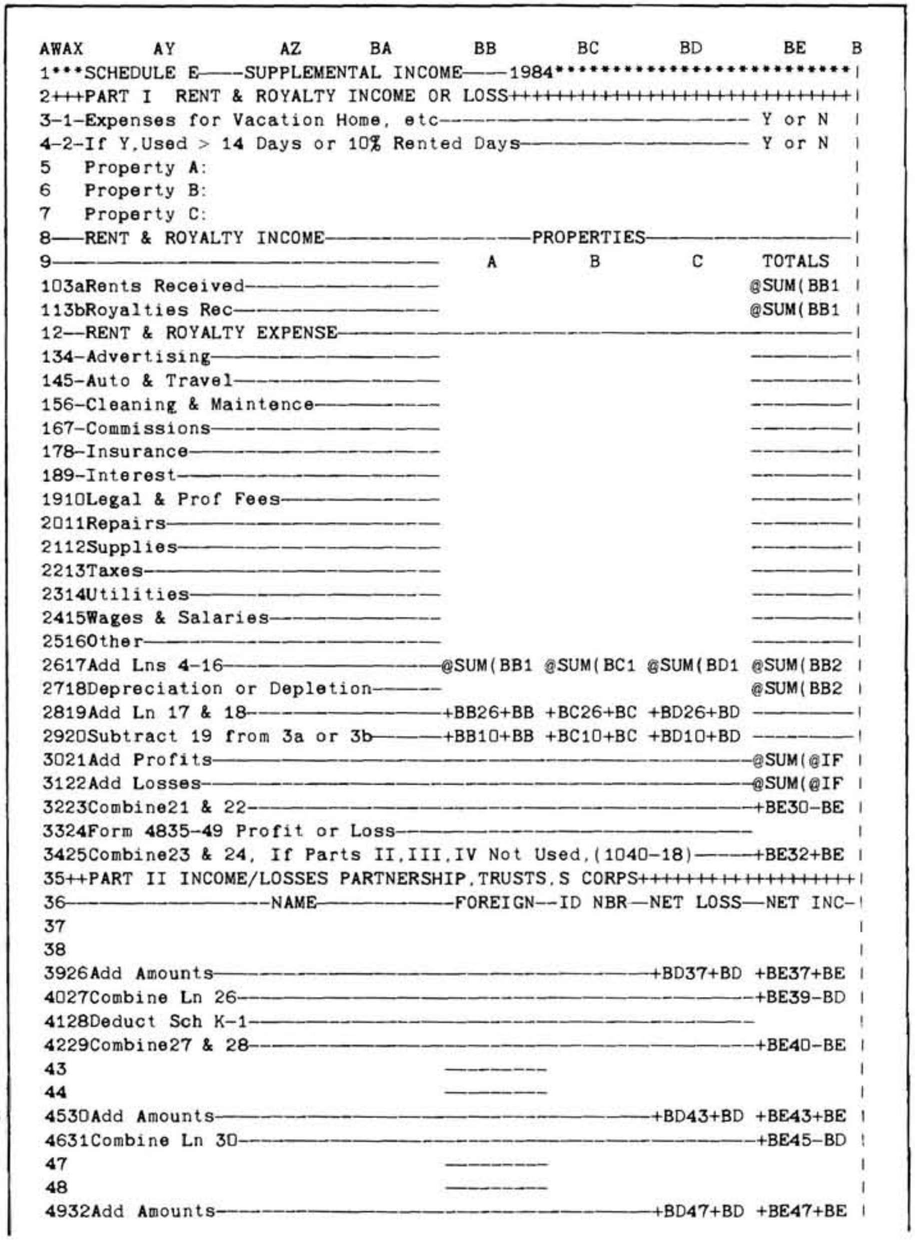
```
N  O         P              Q              R             S  T
73***SCH SE---COMPUTE S/S SELF-EMPLOYMENT TAX****************|
74+++PART I REGULAR COMPENSATION+++++++++++++++++++++++++++|+  |
751---Net Profit/Loss Sch F/K-------------+D24+D25          <---|
762---Net Profit/Loss Sch C/K-------------+L81              <---|
77+++PART II OPTIONAL COMPUTATION++++++++++++++++++++++++++++++|
783---Maximum Inc This Part---------------          1600 <---|
794---Farm Method(F-31/87 or K-18a)<1600-@MIN(R79,1600)  <---|
805---Subtract Ln #4 from 3-------------1600-R79         <---|
816---Non-Farm Method < 1600------------@MIN(R81,1600)   <---|
82+++PART III COMPUTATION OF S/S TAX++++++++++++++++++++++++++|
837---Enter Amt Part I-1/II-4-----------+R75+R79         <---|
848---Enter Amt Part I-2/II-6-----------+R76+R81         <---|
859---Add 7 & 8 if < 400 = 0 -----------@IF(R83+R84>400 <---|
8610-Max Earnings for Tax---------------          37800 <---|
8711aTotal S/S Wages--------           --------------------|
8811bUnreported Tips(4137)---          --------------------|
8911cAdd Lns #11a and 11b--------------+Q87+Q88         <---|
9012-Subtract Ln 11c from 10-----------37800-R89        <---|
9113-Enter Smaller 9 or 12 or 37800-----@MIN(R85,R90,37 <---|
9214-Self-Employment Tax(1040-51)-------+R91*0.113      <---|
93--------------------------------------------------------------|
94                          FIGURE -10B-                      |
```



```
AMAN      AO         AP       AQ       AR        AS       AT       AU    AV
1****SCHEDULE D---CAPITAL GAINS & LOSSES---1984************************|
2++++PART I SHORT-TERM GAINS & LOSSES--HELD 1 YEAR OR LESS++++++++++++|
3----Describe PropDate AcqDate SldGross PrcCost+Exp-  LOSS      GAIN  |
4 1a                                                                  |
5 1b                                                                  |
6 1c                                                                  |
7 1d                                                                  |
8 2--Frm 2119-7/11-------------------------------------------          |
9 3--Frm6252-21/29--------------------------------------------          |
104--Ptnshp,S Corp------------------------------------                 |
115--Add 1,2,3,4-------------------------------@SUM(AT4 @SUM(AU4 |
126--Combine 5 Loss & Gain-----------------------------------+AU11-AT |
137--Loss Carryover---------------------------------------------       |
148--Combine 6 & 7----------------------------------------------+AU12-AU |
15+++PART II LONG-TERM GAINS & LOSSES--HELD > 1 YEAR+++++++++++++++++++|
```



```
7248-Add 42 thru 47-------------------------------------------@SUM(AU6 |
73----------------------------------------------------------------------|
74                          FIGURE -11B-                              |
```



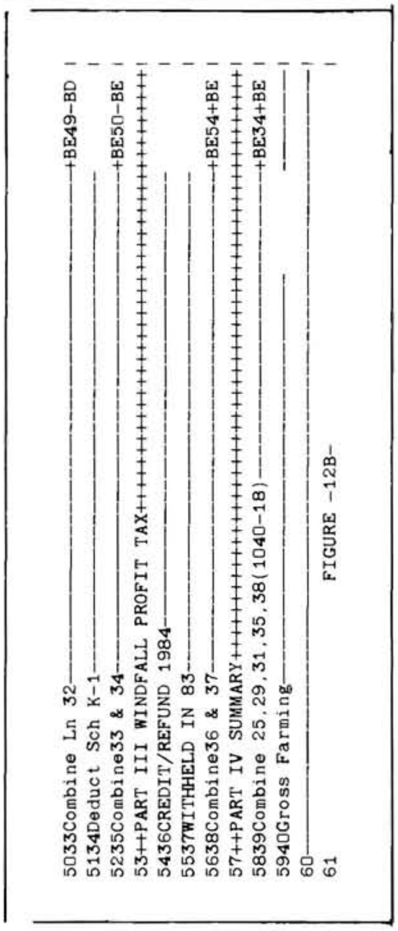
```
AWAX      AY          AZ       BA        BB        BC        BD        BE    B
1***SCHEDULE E----SUPPLEMENTAL INCOME----1984**************************|
2+++PART I  RENT & ROYALTY INCOME OR LOSS++++++++++++++++++++++++++++++|
3-1-Expenses for Vacation Home, etc---------------------------- Y or N  |
4-2-If Y,Used > 14 Days or 10% Rented Days--------------------- Y or N  |
5   Property A:                                                        |
6   Property B:                                                        |
7   Property C:                                                        |
8---RENT & ROYALTY INCOME-----------------PROPERTIES--------------------|
9------------------------------------      A          B          C      TOTALS  |
103aRents Received-------------------                            @SUM(BB1 |
113bRoyalties Rec--------------------                            @SUM(BB1 |
12--RENT & ROYALTY EXPENSE----------------------------------------------|
134-Advertising----------------------                            ---------|
145-Auto & Travel--------------------                            ---------|
156-Cleaning & Maintence-------------                            ---------|
167-Commissions----------------------                            ---------|
178-Insurance------------------------                            ---------|
189-Interest-------------------------                            ---------|
1910Legal & Prof Fees----------------                            ---------|
2011Repairs--------------------------                            ---------|
2112Supplies-------------------------                            ---------|
2213Taxes----------------------------                            ---------|
2314Utilities------------------------                            ---------|
2415Wages & Salaries-----------------                            ---------|
2516Other----------------------------                            ---------|
2617Add Lns 4-16---------------@SUM(BB1 @SUM(BC1 @SUM(BD1 @SUM(BB2 |
2718Depreciation or Depletion-------                             @SUM(BB2 |
2819Add Ln 17 & 18-------------+BB26+BB +BC26+BC +BD26+BD ---------|
2920Subtract 19 from 3a or 3b-------+BB10+BB +BC10+BC +BD10+BD ---------|
3021Add Profits--------------------------------------------------@SUM(@IF |
3122Add Losses---------------------------------------------------@SUM(@IF |
3223Combine21 & 22-----------------------------------------------+BE30-BE |
3324Form 4835-49 Profit or Loss-----------------------------------         |
3425Combine23 & 24, If Parts II,III,IV Not Used,(1040-18)------+BE32+BE |
35++PART II INCOME/LOSSES PARTNERSHIP,TRUSTS,S CORPS+++++++++++++++++++|
36------------------NAME---------------FOREIGN--ID NBR--NET LOSS--NET INC-|
37                                                                     |
38                                                                     |
3926Add Amounts-------------------------------------------+BD37+BD +BE37+BE |
4027Combine Ln 26----------------------------------------------+BE39-BD |
4128Deduct Sch K-1--------------------------------------------         |
4229Combine27 & 28---------------------------------------------+BE40-BE |
43                                  ---------                           |
44                                  ---------                           |
4530Add Amounts-------------------------------------------+BD43+BD +BE43+BE |
4631Combine Ln 30----------------------------------------------+BE45-BD |
47                                  ---------                           |
48                                  ---------                           |
4932Add Amounts-------------------------------------------+BD47+BD +BE47+BE |
```

Games & Spreadsheet Contest Winners Announced

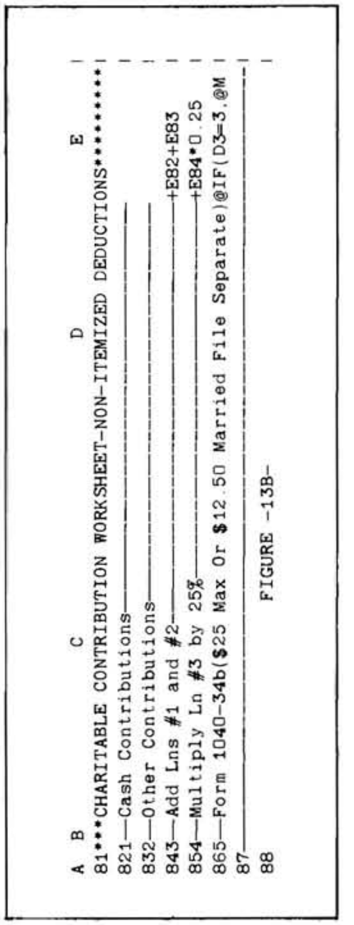
The votes are in and the winners of the Games and Spreadsheet contests are as follows:

Games - **Morris Proctor** of Memphis, Tennessee for his program **WORM**.
Spreadsheet - **Robert Hassard** of Walnut Creek, California for his program **INCTAX84**.

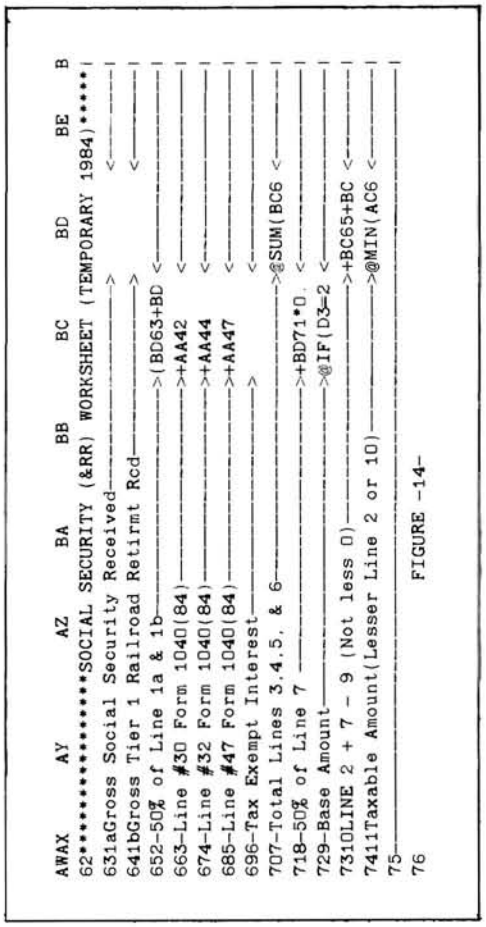
Congratulations for a job well done! Morris and Bob will each receive a $1,000 Gift Certificate for their efforts. Also, a Special Thanks goes out to all who participated in these contests.



```
5033Combine Ln 32-----------------------------------------------+BE49-BD |
5134Deduct Sch K-1-------------------------------------------           |
5235Combine33 & 34----------------------------------------------+BE50-BE |
53++PART III WINDFALL PROFIT TAX+++++++++++++++++++++++++++++++++++++++++!
5436CREDIT/REFUND 1984-------------------------------------------        |
5537WITHHELD IN 83-----------------------------------------------        |
5638Combine36 & 37----------------------------------------------+BE54+BE |
57++PART IV SUMMARY++++++++++++++++++++++++++++++++++++++++++++++++++++++|
5839Combine 25,29,31,35,38(1040-18)-----------------------------+BE34+BE |
5940Gross Farming-------------------------------------      -------------|
60-----------------------------------------------------------------------!
61                          FIGURE -12B-                                 |
```



```
A  B                 C                           D                  E    |
81***CHARITABLE CONTRIBUTION WORKSHEET-NON-ITEMIZED DEDUCTIONS*********|
821--Cash Contributions----------------------------------------          |
832--Other Contributions---------------------------------------          !
843--Add Lns #1 and #2-----------------------------------------+E82+E83  |
854--Multiply Ln #3 by 25%-------------------------------------+E84*0.25 |
865--Form 1040-34b($25 Max Or $12.50 Married File Separate)@IF(D3=3,@M   |
87-----------------------------------------------------------------------|
88                        FIGURE -13B-                                   |
```



```
AWAX      AY          AZ        BA       BB        BC        BD        BE    B
62****************SOCIAL SECURITY (&RR) WORKSHEET (TEMPORARY 1984)*****|
631aGross Social Security Received---------------->            <-------|
641bGross Tier 1 Railroad Retirmt Rcd------------->            <-------|
652-50% of Line 1a & 1b------------------->(BD63+BD <------------------|
663-Line #30 Form 1040(84)---------------->+AA42    <------------------|
674-Line #32 Form 1040(84)---------------->+AA44    <------------------|
685-Line #47 Form 1040(84)---------------->+AA47    <------------------|
696-Tax Exempt Interest------------------->         <------------------|
707-Total Lines 3,4,5, & 6------------------------->@SUM(BC6 <---------|
718-50% of Line 7 ------------------------>+BD71*0. <------------------|
729-Base Amount--------------------------->@IF(D3=2 <------------------|
7310LINE 2 + 7 - 9 (Not less 0)------------------->+BC65+BC <----------|
7411Taxable Amount(Lesser Line 2 or 10)----------->@MIN(AC6 <----------|
75---------------------------------------------------------------------|
76                           FIGURE -14-
```

INTELLIBURNER

EPROM–EEPROM–MICROCOMPUTER PROGRAMMER

UNIVERSAL PROGRAMMING CAPABILITIES AT AN AFFORDABLE PRICE

- **Ultra Fast Programming – 2716's in 16 Seconds**
- **Programs & Verifies 8K thru 256K Single Voltage EPROMs**
- **Erases, Programs & Verifies 2815 & 2816 EEPROMs**
- **Programs & Verifies 8748 and 8751 Series MICROCOMPUTERS***
- **Programming Chracteristics Selected by Convenient Personality Jumper Plug (DIP Header)**
- **Program, Verify, Status, & Diagnostic Display with Tricolor LED**
- **Serial Interface — 3, 4, or 5 wire — 1200 to 19200 Baud**
- **Supports XON/XOFF and READY/BUSY Protocols**

NO SPECIAL SOFTWARE REQUIRED. Transfer disk files (Intel Hex Format) to EPROM with your system's line printer or modem software. Transfer EPROM contents to disk file in Intel Hex Format with your system's modem software. Or use the supplied software ** to transfer any binary or ASCII file to/from EPROM.

PROGRAMS:

2758	2716	27128	27C16	8741*	8748*
2516	2732	27128A	2815	8742*	8749*
2532	2732A	27256	2816		8751*
2564	2764	68764	2817	X2212*	8755*

*Requires low cost personality adapter

SOFTWARE AVAILABLE FOR: **

CP/M systems on 8" SSSD - many 5¼ formats
TRS-80 Model I & III TRSDOS
Heath H8/H89 HDOS & CP/M
ZENITH Z90 & Z100 CP/M - Z100 ZDOS
KAYPRO II/IV
IBM PC-DOS - many MS-DOS systems

IntelliBurner Programmer with Software	$ 299.00
RS-232 Interconnect Cable	12.00
Intelliburner PC Boards, EPROM, Plans & Software	99.00

Low Cost "DumBurner" serial programmers harness the power of your personal computer with the supplied software for full programming capabilities:

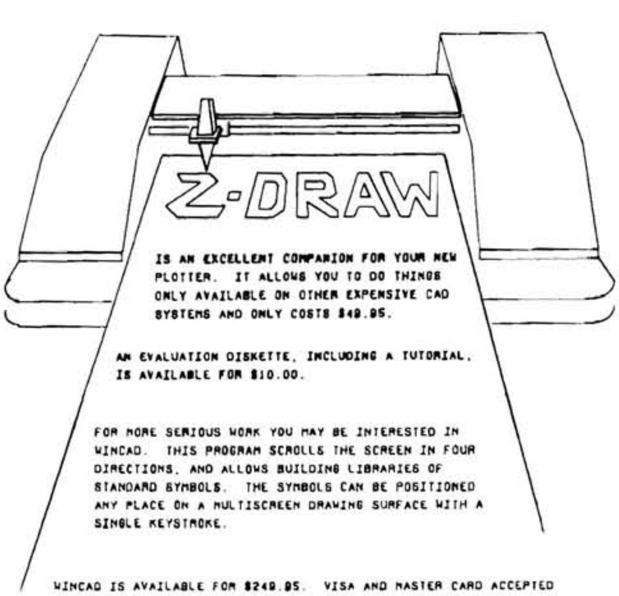
DumBurner II Programmer for 28 Pin and 24 Pin EPROMs and EEPROMs with Software	$ 199
DumBurner II Bare PC Board, Plans & Software	59
16K/32K DumBurner for 24 Pin EPROMs with Software	149
16K/32K DumBurner PC Board, Plans & Software	39

ROSS CUSTOM ELECTRONICS
1307 Darlene Way - Suite A12
Boulder City, Nevada 89005
PHONE (702) 293-7426

! ! NOW SHIPPING with ALL 'BURNERS ! !
TOOLKIT Software for Editing EPROM Information
Plus HEX/MOTOROLA 'S'/Binary File
Conversion Utilities

Ultra Violet Products EPROM Erasers
Model DE-4 - Holds 8 EPROMS - Special Prices

Add $3 Shipping & handling ($2 Bare Boards). C.O.D.'s accepted. Foreign Orders add required postage. Specify Environment and Media Requirements. H8, H89, Z100 are TM Heath/Zenith. CP/M is TM Digital Research. TRS-80 Model III is TM Tandy Corp. IBM-PC and PC-DOS are TM IBM. MS-DOS is TM Microsoft.



Z-DRAW

IS AN EXCELLENT COMPANION FOR YOUR NEW PLOTTER. IT ALLOWS YOU TO DO THINGS ONLY AVAILABLE ON OTHER EXPENSIVE CAD SYSTEMS AND ONLY COSTS $49.95.

AN EVALUATION DISKETTE, INCLUDING A TUTORIAL, IS AVAILABLE FOR $10.00.

FOR MORE SERIOUS WORK YOU MAY BE INTERESTED IN WINCAD. THIS PROGRAM SCROLLS THE SCREEN IN FOUR DIRECTIONS, AND ALLOWS BUILDING LIBRARIES OF STANDARD SYMBOLS. THE SYMBOLS CAN BE POSITIONED ANY PLACE ON A MULTISCREEN DRAWING SURFACE WITH A SINGLE KEYSTROKE.

WINCAD IS AVAILABLE FOR $249.95. VISA AND MASTER CARD ACCEPTED

ASK ABOUT Z-DRAW AND WINCAD AT YOUR LOCAL HEATHKIT ELECTRONIC CENTER.

Redwood
Development
7 Redwood Dr.
Butte, MT 59701
(406) 494-5610

Z-DRAW AND WINCAD WILL RUN ON Z-100 AND Z-150 COMPUTERS. WHEN ORDERING PLEASE SPECIFY.

The Software Toolworks presents: MYCALC™

CAR PAYMENTS. BANK STATEMENTS! TAXES... ARRGH!
YO, FREDDIE. STILL GLAD YOU FIRED THE ACCOUNTANT?
BUDGETING. ACCOUNTING. TAXES... ARRGH!
THIS IS MYCALC™, FREDDIE. IT'S A SPREADSHEET. AND FOR SIXTY BUCKS IT'LL SOLVE YOUR PROBLEMS.
COMES WITH TEMPLATES FOR IRS FORMS 1040 & SCH. A
TWENTY THREE HELP SCREENS! THIS IS EASY, I MEAN MY BUDGET'S ALMOST DONE.
NICE WORK, FREDDIE. NOW PLOT MY CATNIP FUTURES.
MYCALC IS $59.95. IT'S AVAILABLE FOR CP/M (R) AND HDOS (64K) AT MOST HEATH/ZENITH DEALERS. WRITE OR CALL FOR A FREE 41 PRODUCT CATALOG AND NEAREST DEALER.
A SOFT BUY IN A HARD WORLD!

Full featured MyCalc with sort, bar graphs, multiple files and more is ideal for financial planning and budgets. It sells for $59.95 from The Software Toolworks, 15233 Ventura Blvd., Suite 1118, Sherman Oaks, CA 91403 (818) 986-4885.

$ FINANCIAL TOOLS $

FOR YOUR HEATH/ZENITH COMPUTER

- **LOAN-MASTER™** $29.95 CP/M-80 $49.95 MS-DOS
  Flexible loan amortization schedules provide you with the information you need to select the best terms for home and auto loans.
- **INVESTMENT-MASTER™** $29.95 CP/M-80 $49.95 MS-DOS
  Provides lump sum and annuity figures that help you manage periodic savings plans, mutual funds, and IRAs.
- **FUND-MASTER™** $39.95 CP/M-80 $59.95 MS-DOS
  Stock market investment aid. You benefit by knowing when to buy or sell and how much to buy/sell.
- **DEPRECIATION-MASTER™** $149 CP/M-80 $199 MS-DOS
  Comprehensive fixed-assets analysis system. Allcws for book and tax accounting of your business assets. Requires Microsoft BASIC Interpreter.

Dealer Inquiries Invited ★ Quantity Discounts Available
ORDER NOW . . . CLIP THE COUPON AND SEND

YES, please rush me the following software!

□ LOAN-MASTER □ INVESTMENT-MASTER □ FUND-MASTER
□ DEPRECIATION-MASTER □ FREE Catalog of Products
Operating System: □ HDOS □ CP/M-80 □ ZDOS □ MS-DOS

NAME ______________________
ADDRESS ______________________
CITY ____________ STATE ______ ZIP ______
PHONE (______) ______________________
COMPANY ______________________
□ CHECK ENCLOSED □ Send COD (Add $2.00)
□ COMPANY P.O. # ______________
□ VISA □ MasterCard Account # ______________ Exp. Date ______
SIGNATURE ______________________

FOR FASTER DELIVERY CALL
906-249-9801

Add $3.00 shipping/handling for all orders and 4% sales tax for Michigan residents.

Send to: **GENERIC COMPUTER PRODUCTS, INC., POB 790, Dept. 385R**
Marquette, Michigan 49855

Implementing A Hard Disk On The H/Z-100

William M. Adney
P. O. Box 13186
Arlington, TX 76094

With the availability of some of the new integrated software and the continued use of the data base packages, it's clear that high capacity drives, usually in the form of a hard disk, will no longer be a luxury...they will be required. I've heard a lot of interesting comments on the implementation of a hard disk on the H-100, and I found that I needed the capacity to finish the MS- DOS FlipFast book. The reason for that is that I use Star Index to create the Error Message Cross Reference List, and Star Index creates another file which is the same size as the original during its processing. The complete manuscript file for the book was about 700K which translates to about 350 pages on my H-25 printer. That includes all of the typesetting codes so that the final MS-DOS book is 256 pages. The increase in size is also the reason for the slight price increase over the CP/M-80/85 book.

As I mentioned last time, I bought the hard disk at the HUG Convention in Chicago. I had three selections to choose from: the standard HS-217 upgrade kit (11 megabyte), a 26 MB, and a 43 MB drive. The higher capacity drives seem to be a choice made by some of the Heathkit stores because some stores do not carry them. Why? Those high capacity drives are not actually "certified" for use with the H-100 system. Although I haven't had any trouble with the new drive, there is at least one good reason for the lack of "certification" on the high capacity drives, and we'll talk about that later.

Although the 43 megabyte drive was clearly the most megabyte per buck, I chose the 26 megabyte since it was large enough to meet my storage requirements (both now and in the future), and it was small enough to be "relatively easy to back up". In my mind, the 43 megabyte was too big to be backed up conveniently. It required too much time and too many disks.

Ironically, this particular column has been planned for about six months. The move to our new house in Texas has interfered with my writing schedule, but the MS-DOS book took first priority. In any case, I hope that this article answers Ken Goto's request for information which was on page 5 of the December issue of REMark.

Why Is It Called A Winchester?

Have you ever wondered why the documentation talks about a Winchester Supplement or Winchester Utilities? Believe it or not, the name originated at IBM. Back in the days when the hard disk was being developed, IBM was working on one which had a lot of storage...30 megabytes above and 30 megabytes below. And so it got the name 30-30. Those of you who are hunting enthusiasts will recognize that 30-30 is also the name of a popular caliber hunting rifle made by Winchester. It seemed a natural to call the new hard disk a "Winchester", and the name has stuck through the years. It's interesting to note that Winchester technology is nothing new. What is new is the size versus the storage capacity. I have a 5.25-inch Winchester drive that has 26 megabytes (unformatted) of storage capacity. It's interesting to know that mainframe computer disks (e.g. IBM 3380) are in the gigabyte (1,000 megabytes) capacity.

As I was writing the IBM PC-DOS FlipFast book, I found it somewhat unusual that IBM does not called their hard disk a Winchester. It's called a fixed disk. As a matter of fact, PC-DOS has a command called FDISK which is essentially equivalent to the Heath/Zenith PART command. Interesting that IBM has apparently abandoned the name that they gave to the technology. For my part, I like to use the term "hard disk" since it seems to provide a nice contrast when one is also discussing floppy disks.

Installing the hard disk

The kit consists of 3 basic boxes and the usual number of manuals. The first box contains the faceplates, mounting brackets, and miscellaneous hardware. The second box contained the manuals, Z-217 disk controller board, and the data separator board. And the final box contained the hard disk itself.

Conversion kits are available for all of the H-100 styles. That includes the Low-Profile and the All-In-One with either the full height or half height drives. If you have an older system, you'll need the HS-217 for the full height drives. Newer systems with the half height drives require the HS-217A. The only difference between the two is the panels that are furnished with the kits.

Installing the hard disk is relatively straight-forward. The Heath manuals continue to be excellent in that respect, and I had no problems. The overall hardware modification took me about 6 hours, but I spent considerable time checking some things. The first part of the manual requires that you disassemble your computer to get to the main circuit board. Most of the work is easy. However, you may have to make some modifications to the main circuit board.

Do You Need to Modify The Main Circuit Board?

The first major thing that you need to look at is the part number on the circuit board. If the part number on your circuit board is 85-2653 (disregard any other number that may follow this number, such as 85-2653-1), you will have to modify the main circuit board. If you don't have that part number on the main circuit board for the All-In-One, you can then proceed with the reassembly of the unit.

At this point, the instruction manual has a slight difference depending on whether you have the Low-Profile or the All-In-One. For the Low-Profile, you are also instructed to connect the leads of an ohmmeter (set to the lowest range) from U130 pin 10 to U225 pin 3. If the ohmmeter indicates anything other than a short circuit (very close to zero ohms), you will have to modify the main circuit board. Otherwise, the work on the main circuit board consists primarily of replacing some of the IC's (ROM's).

Modifying the main circuit board

There are four modifications that need to be made to the main circuit board which involve cutting one trace and installing three jumpers. Heath has thoughtfully included a magnifying lens with the kit to make it easier to see the trace. Since cutting a trace is a rather permanent modification, it's a good idea to verify the location several times before you do any cutting. That trace is located at U168 pin 11.

After the trace has been cut, three jumper wires are installed. The first is from U152 pin 1 to U130 pin 8. The second is from U130 pin 9 to R104. The last jumper is from U130 pin 10 to U225 pin 3.

Although this is quite clear in the Heath manual and the pictorials, it's more than a little difficult to describe it without them.

Installing the hard disk

At this point, you will reassemble your system to the point where the disk drives are installed. If you had two full height drives, you will move the old A drive to the B drive slot and set the old B drive aside. As usual, you have to cut a programming plug for the hard disk. Then you have to add a couple of brackets to the drive unit to support the data separator board which mounts on the top of the drive. The Z-217 hard disk controller is inserted into any of the open S-100 slots. Since I had a "non-standard" disk size (26 megabytes), I left the format enable jumper in the "enable" position because I had to run PREP. According to the factory manuals, you will not have to run PREP on the standard (HS-217) hard disks because that's already done for you at the factory. From then on, it's a matter of reinstalling the drives, plugging in cables, and mechanical assembly.

Setting up the hard disk

It took about 3 1/2 hours to run PREP on the 26 megabyte disk. Next is the partitioning with the PART command. Use some thought before you set up your partitions because any changes to partition size will require that you backup and reload the entire hard disk. Depending on the type of data that you intend to keep on the hard disk, you might want to consider setting up CP/M partitions based on some approximations to the size of your backup media. For example, I decided that I wanted to use about a 1 megabyte partition for writing the FlipFast books. I chose that size since I can back up the entire partition to one 8-inch floppy disk (980 K) using the PIP command. Strictly a choice of convenience. For MS-DOS, I've allocated approximately a 5 megabyte partition. Since I use the subdirectories (each one has its own backup disk), I'm very careful to keep the total space usage under the 1.2 megabyte limit for the 8-inch disk. You could accomplish the same thing in CP/M by careful assignment of user numbers, but I don't use them.

A hint on setting up the partitions for your hard disk...ALWAYS keep your complete set of operating system (CP/M or MS-DOS) and application (e.g. WordStar, dBase II, Lotus 1-2-3) on a single partition. While that sounds like putting all of your eggs in one basket, it makes the backup process MUCH easier. How often do you change the software compared to the data files? Not very often, right? Once you have made a backup of the software partition (drive E in MS-DOS), you won't have to worry about another backup until you add some software to that partition. A periodic backup of the data partitions (at least monthly - weekly for businesses or any data intensive applications).

You may also want to do some experimenting with the BACKUP and RESTORE commands. These commands, with their various options, provide an incredible amount of flexibility for backing up a hard disk. They are far superior to anything else I've seen, including the ones available in IBM PC-DOS 3.0. If they're so good, how come I don't use them? That's just a personal thing...I established a backup procedure several years ago, and I guess I'm just too fixed in my system now. That system has saved me more than once, and I'm reluctant to change it. It's something like the old saw that says: "If it works, don't fix it". I use AutoDex (see REMark May 1984) to back up each file from my hard disk to a floppy every time I update a file. I also periodically backup the entire partition.

Back to setting up the hard disk. After you've made the partition assignments, I recommend formatting ALL partitions, even the ones assigned to CP/M with the MS-DOS FORMAT program. As you're doing that, keep a log of the number of bad sectors (divide the value of bytes in bad sectors by 512) in each partition with the date. That will give you some idea of the status of your hard disk. Don't be too surprised if you end up with 30 or so bad sectors for the entire hard disk. EVERY hard disk has bad sectors.

I recommend that you run the PSC command before you run PART so that you can print the screen showing the final partition names and sizes.

Loading the Hard Disk

The first thing you'll need to do is load the operating system. As usual, all of the CP/M partitions must be formatted using the appropriate FORMAT command. For CP/M-85, you must use the MVCPM217 command before the SYSGEN in order to be able to boot the hard disk from the Z-217 disk controller. A similar process is required for CP/M-86 using LDCOPY with BOOT217.

Loading MS-DOS to a hard disk is a little easier. You can use the FORMAT command (i.e. FORMAT E:/S/V) with the /S (System) option to verify the disk and transfer the operating system as a one step process.

I recently received a letter from Gil Kendrick asking how you update to MS-DOS 2.0 when you're already running Z-DOS. He wanted to know if it was necessary to completely unload the hard disk in order to add the new MS-DOS to it. Unfortunately, the answer is yes. It's not as easy as you might think. For my money, the SYS command gets my vote for the most useless command in the operating system. This command will NOT transfer the operating system unless the system files (IO.SYS and MSDOS.SYS) are the same size or less than the system files currently on the disk. What this means to you is that you cannot simply SYS any of your current Z-DOS disks because the version 2 operating system is bigger. You will have to get some extra blank disks, format them with version 2 FORMAT, and transfer all files to the newly formatted disks. If you have a hard disk, you'll have to backup the entire partition, format it, and reload all files back to that partition. That problem is part of the boot loader and isn't something that's easily fixed. The boot loader isn't smart enough to figure out how to load those files when they're bigger. For what it's worth, PC-DOS (all versions) works the same way.

If you decide to by a hard disk for your H-100, I recommend that you also buy about 3 boxes of floppies so that you'll have a complete way to back up all partitions. That should give you enough spares to do some other things too.

Now that you have your operating system(s) on the hard disk, you are ready to transfer files from your current floppy disks to the hard disk. Do so with the PIP command (CP/M) or the COPY command (MS-DOS). In this particular case, I recommend using the verify ([V] for PIP and /V for COPY) to verify the integrity of the file transfer.

Now that you have all of your software and data on the hard disk, you are ready to begin working with it, right? Absolutely WRONG! The very next thing you should do is back up each partition to its own set of floppies, label them, and keep them near your system for emergencies. For CP/M, I recommend the use of a command in the form of:

```
BACKUP C:PART1=A:*.*[V]
```

Assuming that your current drive is the hard disk (drive A), that command will back up your files to the 5.25-inch drive C and verify the backup. You will be prompted for the date of the back up by the BACKUP program.

For MS-DOS, I recommend the use of a command in the form of:

```
BACKUP *.* A:PART1/V
```

I have again assumed that your current drive is the hard disk (drive E), and you will be backing up your files to the 5.25-inch drive A with verification.

Note the use of the partition number in each command makes it easy to tell the source of the backup, even if you decide to change operating systems. If you have critical (e.g. business) data on your hard disk, I also recommend that you keep an extra set of backup disk at an off-site location for protection from fire and natural disasters.

Standard Operating Procedure

Before you start thinking that the hard disk is a panacea for all your floppy disk storage problems, you had better establish a routine for backing up your hard disk partitions. Remember that a hard disk is a special kind of mechanical device, and mechanical devices do fail. Even if the hard disk is perfect, little things like power failures can absolutely destroy all data on the disk. I recently had that happen to me. Apparently the directory on the hard disk was being written when the power failed. The result was that I lost everything in that partition because the directory went to outer space. I make it a practice to save all data on my hard disk about every 15 minutes. About every two hours or whenever I feel like taking a short break, I back up that file to a floppy disk. Even though that power failure completely wiped out the disk directory, I only lost about 20 minutes worth of work. The worst that could have happened was a loss of about two hours' work.

I'd like to stress the importance of taking regular backups. That applies to floppy disks as well as hard disks. Although the chances of losing a complete disk directory in a power failure are extremely small, there are many other ways to lose files. Of course I suppose that none of you have ever typed ERA *.* when you really meant ERA B:*.*. Or DEL *.BAT when you meant DEL *.BAK. What happens when someone (like me) spills a cup of coffee on a floppy disk? We'll deal with all of that and more in one of my next columns: "Power Glitches and Sloppy Floppies".

The Bad Sector Table

For those of you who have one of my FlipFast books, you'll find a rather obscure error message for the PREP command: "Bad sector count exceeded for this drive". The current limit for the bad sector table is 169 bad sectors which translates to 86,528 bytes assuming a 512 byte sector. As I mentioned last time, the first drive I received had a problem which was corrected in a very timely manner by the Heathkit store. The point is that I could not use the first drive because of that bad sector limit. The PREP program kept stopping during the process because that limit had been exceeded.

Before you get too worried about bad sectors on a hard disk, consider that a certain number of bad sectors are normal for ANY hard disk. That includes the disks used on mainframe computers. As you might expect, mainframe computers have always had utilities for "locking out" the bad sectors since that is a normal consequence of the media. By the way, you don't see bad sectors occurring on floppy disks very often because of the data density and the way that they're manufactured. Rough treatment of floppy disks is a good way to create bad sectors. At this point, I have over 300 floppy disks, and I haven't had any bad sectors occur on any of the brand name disks. More on that later.

An interesting observation about the bad sector table is that Zenith has established the 169 limit on the number of bad sectors. That number is a result of some technical considerations that each bad sector table entry takes 3 bytes for a total of 507 bytes. The last entry in the table always contains 000 (3 bytes for the end of the table) plus 2 additional bytes that are reserved for future expansion. As it turns out, 512 bytes is exactly one sector. A rather complete discussion of all of this is contained in the Z-DOS Winchester Supplement beginning on page 11.12.

I brought all of this up because there is at least one piece of software that I think is really needed for the hard disk systems. We need something that reports on the number of bad sectors as well as a list of the address of each one. There are, by the way, two bad sector tables written (the second is a copy of the first for back up), so the program should list both and compare them too. The program should also look at the Superblock entry and identify which bad sectors are in which partition. The only way that I've found to get an idea of that is to back up a partition and format it with the MS-DOS format program which reports the number of bytes in bad sectors. CHKDSK also reports the same thing. Since I know that MS-DOS uses 512 bytes per sector, I can then calculate the number of bad sectors for each partition. That's really a pain because I currently have 10 partitions.

Before we get too much further, I believe that ZDS really needs to look at the partition table size with an eye toward expanding it. The argument is, of course, that the current limit is quite satisfactory for the currently approved HS-217 (H-100's) or HS-317 (H-150/160's) disks. That's true, but a little research will show that the 11 megabyte disk will be (already is?) one of the smallest ones available. I've been reading about 100+ megabyte drives for microcomputers, and with the strides being made in technology today, those will probably be commercially available in a couple of years. While that kind of capacity is probably not needed for home/hobby use, it is something that the business (maybe the military too) world is taking a look at. High capacity drives will be in great demand in the coming years.

What About the Extra Disk Drive?

Now that your hard disk is up and running, you're probably wondering what to do with the extra drive that you have to remove to make room for the hard disk. One option is to put it in the closet and use it for a spare. That doesn't work out too well for me because I've found that having two 5.25-inch drives is very important. At least one reason is that I use the DUP and DISKCOPY programs frequently for various backup copies especially when I send a disk to REMark or to my publisher for a FlipFast book. Another option is to chuck the other full height drive and buy two half height drives. That seems like a lot of money to spend when you already have the two drives.

Studio Computers to the Rescue

Fortunately, I had a Studio Computers catalog, and they have external drive cabinets for the 5.25-inch drives. I called Ray Massa with a couple of questions and ordered a single drive enclosure (SC-100) which was perfect. The cabinet arrived the following week, and it's a close match to the H-100. One of the impressive things was the instruction sheet enclosed with the cabinet. It's a brief, single page description of how to hook up the cabinet, but it tells you all you need to know about the cabinet. The only reason for that comment is that I've ordered some cabinets from a different vendor, and received NO instructions with it. Even if you don't need anything right now, I suggest that you request Studio Computers catalog for future reference. I recommend them as an excellent source for just about anything (hardware or software) that you'll need. You'll find that they have very reasonable prices, and they provide excellent delivery.

Balancing Your Checkbook?

One of the most common questions that I get asked by people who don't have a micro yet is: "What can I use a microcomputer for ?". Right after that, people want to know what kind of software is available for checkbook balancing since that seems to be the most commonly thought of use for a micro.

Newline Software's Check Processor is one of the answers to the checkbook balancing problem. It has a number of features that make it much easier to use than writing a spreadsheet for balancing a checkbook, and includes extensive use of the H/Z-100's function keys. I've got the MS-DOS version which runs under 2.0 on the H-100, so that's what we'll be looking at. The Check Processor is also available for the Z-150/160's and the IBM PC.

Check Processor includes 2 programs: CHKPRO and CHKLIST. CHKPRO is the actual checkbook processor which allows you to enter debits and credits for checking, savings, and credit cards. CHKLIST provides special listings of the files created with CHKPRO. There is no size limitation on the files except for the space available on the disk drive.

Documentation for this package includes 6 pages of instructions which can be read very quickly. Since the programs use function keys, which are labelled on the 25th line, virtually all of the functions are self-explanatory so you don't have to remember a lot of complicated commands. CTRL-C is used to abort processing without saving the input data, otherwise the use of F4 (menu) or F5 (exit to MS-DOS) results in saving the data to the current file.

One of the most interesting features of the program is that you can enter a single letter in the description. For example, I for interest, followed by a carriage return, causes the description field to expand to the entire word "INTEREST". Similarly, W for withdrawal, D for deposit, and C for credit may be input. Another good feature is that by placing an X in a special column, reconciliation of the account is done automatically, and the reconciled balance always appears at the bottom, right corner of the screen.

I've used the program for a few months now, and in general, I like it. Bill Langlais did a real nice job on the package, used the H-100 function keys, and generally made the programs easy to use. The only real problem that I've had has to do with the hard disk in my H-100. All of the programs and files for the Check Processor are in an MS-DOS subdirectory called "\MONEY". Although the basic program, CHKPRO, works well; the program which provides a list of the checking account, CHKLIST, always wants to find something on drive A. From what I can tell, it's only looking for a disk in the drive or an MS-DOS "Not ready" error is displayed. I've even used a CP/M-85 disk in the A drive, and that satisfies the requirement since CHKLIST apparently does not try to read it. Ron Rocheleau tells me that some enchancements are being made to Check Processor, and that "drive A" problem is being fixed. Those changes should be available by the time you read this. Other changes are also being made which should make the programs even better. For those of you who already have the program, there will be a modest update charge...check with Newline Software on their update policy. Check Processor is an excellent program, and I do like it. It's recommended. I've also seen that the program seems to be available at many of the Heathkit stores, as well as by mail order at the address shown at the end of the column.

Interrogator by Dysan

One of the most unique products that I saw at the HUG Convention last year was a disk drive diagnostic program called Interrogator by Dysan. Aside from being a most useful package, it has a rather spectacular display on a color monitor although it obviously works with a monochrome monitor too. Of course the Dysan folks had the color monitor since Interrogator provides an extremely eye-catching display. And the best part is that the have an Interrogator program customized for the H-100.

Why would you buy a diagnostic program for your disk drives? I mentioned earlier that you should make backup copies of your disks due to unforeseen problems. The Interrogator is another tool that can help you spot potential problems BEFORE you lose any data. Aside from that, it's just a neat program. If you have a color monitor, it's spectacular.

The program is completely menu driven and easy to use. Extensive use of the H-100's function keys are one reason that the program is impressive. One of the more interesting aspects of Interrogator is that you boot the system from the Interrogator disk. It's operating system independent since Dysan has developed all of the support for the software which is contained on the disk. After you boot the system, you are prompted to remove the program disk and press any key to continue.

The main menu contains six options, all selected by the function keys: Help, Auto Sequence, Alignment Test, Utilities, Read/Write Test, and Setup Parameters. If you don't want to read the documentation, the Help menu provides just about everything you'll need to know about using the program. Although it's not necessary because everything is set up for instant running of the program, I'd suggest that you review the five Setup Parameters. The Auto Pass/Fail parameters are preset and check all of the physical characteristics of the disk drive such as RPM, centering, and so on. If you're so inclined, you can change any one of these parameters with this menu although I haven't found any need for that. Program parameters include the number of read/write retries, a toggle for sound (I don't care for the beeps), selection of a serial or parallel printer, and a toggle for color. Drive parameters include the default values for 5.25-inch and 8-inch drives including the step rates. And of course you can set up the baud rate and other values for your printer, and make backup copies of the custom configured program for everyday use.

The Auto Sequence test is basically a check of the physical drive characteristics such as centering, alignment, and rpm. The screen provides a display of the results of the test for both sides of the disk, the "accepted values and deviations", and a Pass/Fail indication. This is the overall testing facility. If you find any problems, you should then go to the more detailed tests.

As far as I'm concerned, the Read/Write test is one of the best displays I've ever seen. Sectors for both sides of the disk are mapped on the screen by display blocks about the size of a block cursor as a blank disk is being formatted for the test. Once the test begins, the blocks are filled in an apparently random way as the testing begins.

The Alignment test menu allows you to select the drive for testing and specify the characteristics of that drive. Checks for speed, centering, radial alignment, azimuth, indexing, and hysteresis can be performed. I won't try to explain what each of those tests mean since they're quite well explained in the Dysan manual.

After using this program for a while, I have to say that I'm impressed by its performance and ease of use. The manual that comes with the package also lives up to the high quality that we have come to expect from Dysan.

Those of you interested in this kind of program are probably wondering what kind of price this kind of quality has. To my mind, it's a little steep at $139.00, although I have to admit that it's a nice piece of work. I mentioned this to Elizabeth Dessuge, Product Manager for Dysan, at the HUG Convention. The basic answer is that the cost of disk drive repair is so high that this program will save you money in the long run. In addition, it's difficult to calculate the cost of lost data when a disk drive heads toward outer space. That last comment was mine by the way. Although the version I tested only included the Digital Diagnostic Diskette for the 5.25-inch drive, you can also obtain one for the 8-inch drives. I also understand that Dysan is working on an Interrogator program for our hard disks. More on that when I see it.

Speaking of Dysan

One of the most puzzling things to me is that people will spend all kinds of money for a computer system and software, and then try to save a few pennies on floppy disks. Floppy disks are the absolute worst thing that you should try to cut corners on in your entire system. Why? Think of the investment that you make in time generating the data stored on those disks. That time investment can be incredible over just a couple of years. Since I have a full time job, I simply can't afford the time required to reenter all of the information that I currently have on my floppy disks. Most people have heard that Dysan disks are the best you can buy. ALL of my important information (like the FlipFast books) is at least backed up (from the hard disk) on floppies. I use Dysan disks for all original data that is not on my hard disk too. More importantly, I've never had a problem with Dysan's. Well, almost never. I guess that you can't really count the time I spilled coffee on one of my master disks. Fortunately I had a backup copy. Although the Dysan's cost more than most other brands, I recommend them and use them daily. I'd rather spend a little more for the disks and recognize that as "peace of mind insurance" for my data.

That's not to say that I don't use other brands of disks. Of course I do. I've tried any number of brands including 3M, Verbatim, Memorex, and various disks available through the Heathkit stores. Although I've never had any problems with any of them, I still buy the Dysan disks for the important things. My second choice, because of quality and price, still is 3M. I usually buy single side, double density disks in quantities of 50 to use for sending information for publication.

Because of price, I buy all of those 3M disks at Disk World! Good quality disks with prompt delivery and good prices are the reason. I won't try to quote any prices since Disk World always seems to have a special of one kind or another. For example, the Disk World! ad in the December issue of Byte shows that, for a minimum order of 20 5.25-inch disks, you can get SSDD disks for $1.53 each or DSDD disks for $2.06 each. Write to Disk World! to get their latest prices on various brands.

My last purchase of Dysan disks (8-inch, DSDD) was from Lyben Computer Systems. Good prices and quick delivery seem to be the rule for him. A recommended source for all sorts of computer supplies. Write to Lyben for the latest information on prices. From what I've seen, Dysan prices have become a lot more competitive in the last year, and Lyben has particularly good prices on them.

Hints and Kinks for MS-DOS EDLIN

One of the best kept secrets seems to be the line editor, EDLIN, that comes with MS-DOS. It's not like the CP/M ED program...EDLIN is very easy to use which is primarily due to the use of the function keys for various operations. Those same function keys work on the MS-DOS command line as well as DEBUG. A rather complete description of the function keys is included in the EDLIN section of the manual in case you couldn't find it. If you haven't tried EDLIN, I suggest that you take a few minutes to play with it. Follow the examples in the MS-DOS manual. I use EDLIN to develop BAT files since I usually need them quickly and don't want to wait for WordStar to load itself.

I have found a slight problem with EDLIN however. It's more annoying than anything else, and it becomes most noticeable in a batch file. The symptom is that a double command prompt is displayed after the batch file has completed processing...just like you entered an extra carriage return to the file. As a matter of fact, that's exactly the problem. A little investigation with DEBUG shows that, at the end of the file, EDLIN adds a CR LF (carriage return, line feed - 0D 0A hex) sequence followed by the normal End-of-File character, CTRL-Z (1A hex) which causes the extra prompt. The way around that is very simple...when you get to the end of the file in the Insert mode, enter a CTRL-Z on the last blank line before you return to the EDLIN prompt. The operating system thinks that CTRL-Z is the end of the file which eliminates the double prompt. Another look at the file with DEBUG will show the end of your data followed by a CTRL-Z, CR, LF, and a final CTRL-Z. All you've done is fooled the system into thinking that the End-of-File occurred before it found the CR LF. Try it, it works!

In the Mail

Just thought I'd take a few lines to mention one other thing. If you have a question or a problem (or even a suggestion or complement!) that I can help you with, let me know. If you want a personal answer, please enclose a stamped, self-addressed envelope. Although I don't promise an typewritten reply, I'll answer all letters that include the stamped envelope. If you consider my scrawl an answer, that is. For those of you who are wondering, I will also be talking about software for the Z-150/160 PC series which will also apply to our members who own IBM PC's.

Next Month

Barring any unforeseen developments, I will try to do a little catching up with some things next month. A lot of software needs to be reviewed, and it will be a general column to get caught up on everything. I mentioned unforeseen developments...one possibility is that I will be reporting on the Gemini emulator board. In case you haven't heard about this, the Gemini will allow your H/Z-100 to run IBM PC software. From what I understand, it plugs into the mother board and will be distributed exclusively by Heath Company. Whenever I get one of those jewels, I'll report on the results of my testing. Does it boot IBM PC-DOS? Keep on reading this column to find out.

Products Reviewed

FlipFast Command Guides
CP/M-80/85 .... $12.95
MS-DOS(Z-DOS) $19.95
IBM PC-DOS .... Not Available Yet
Heathkit Stores
S-A Design Books
515 W. Lambert, Bldg. E
Brea, CA 92621-3991
(714) 529-7999

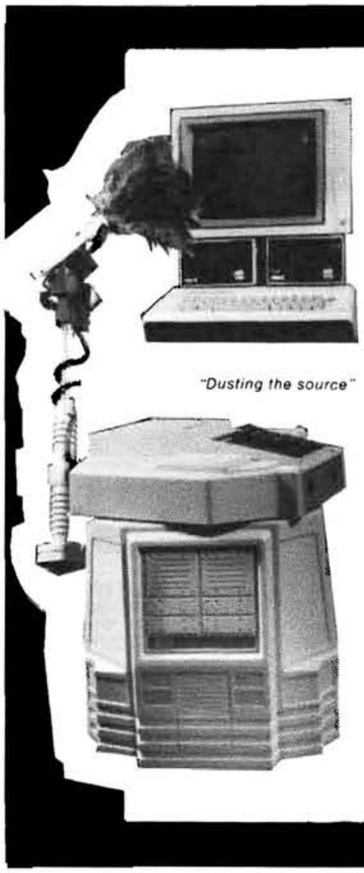
MS-DOS 2.0 (OS-61-8) $150.00
Winchester Upgrade Kit (H-100) $1799.00
Full height drives (HS-217)
Half height drives (HS-217A)
Heath Company Parts Department
Hilltop Road
St. Joseph, MI 49085
(616) 982-3571

Check Processor
H/Z-100 .... $29.95
Z-150/IBM PC $39.95
Newline Software
P.O. Box 289
Tiverton, RI 02878
(401) 624-3322

Interrogator $139.00
Dysan
5201 Patrick Drive
P.O. Box 58053
Santa Clara, CA 95050
(800) 551-9000

Disks and Computer Supplies
Lyben Computer Systems
1250-E Rankin Drive
Troy, MI 48083
(313) 589-3440

Disk World!, Inc.
Suite 4806
30 East Huron St.
Chicago, IL 60611
(312) 944-2788 (Information)
(800) 621-6827 (Orders Only)



ATTENTION ZENITH/HEATH COMPUTER USERS AND SUPPLIERS

MICROSERVICES announces that a new company, **COMPUTER GRAPHICS CENTER, INC.**, has been formed. The goal of **CGCI** is to provide a common source for graphics and other hardware/software products which provide significant advances to computer and video system end users. As part of its business area, **CGCI** will expand the support to the Zenith/Heath community previously handled by MICROSERVICES. All product lines previously handled by MICROSERVICES are being transferred to **CGCI**. MICROSERVICES will continue as a product development oriented company.

Z-100 and **Z-150** series users with graphics interests can acquire a catalog from the address below. We are also interested in hearing from developers who need publication, marketing and distribution support for their products.

COMPUTER GRAPHICS CENTER, INC.
140 UNIVERSITY AVE., SUITE 65
PALO ALTO, CA 94301
PHONE: (415) 851-3414

JOIN US ON OUR JOURNEY TO ADA* VALIDATION

As part of its support to advances in the computer industry, **COMPUTER GRAPHICS CENTER, INC.** has acquired a distributorship from RR Software for their Ada related products. They include the **JANUS/Ada** compiler and a full development system. The present compiler contains many of the features of the full Ada language. Growth to validation is scheduled for the near future.

The IBMpc version of the compiler works on the **Z-100** and the **Z-150** series computers. It is now offered at a truly amazing price of $99.95. The newly revised development system, which includes a complete tool kit, assembler and other products is priced at $900.00.

A low cost disk is available for assisting the **Z-100** user in making high resolution graphics. Software update contracts and site licenses are also available. Send for your copy of the compiler or development system today to learn and apply the computer language destined for tomorrow.

COMPUTER GRAPHICS CENTER, INC.
140 UNIVERSITY AVE., SUITE 65
PALO ALTO, CA 94301
PHONE: (415) 851-3414

Add 3% shipping/handling, $3.00 min. California add 6.5% tax.
*Ada is a trademark of the U.S. Department of Defense.

HEATH/ZENITH 88, 89, 90 PERIPHERALS

16K RAM EXPANSION CARD

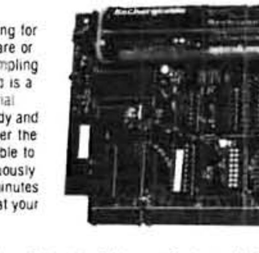
Expand your H/Z 88, 89 RAM Memory to a **FULL 64K** and begin using larger and more powerful programs with our **16K RAM** card.

Fully compatible with: Magnolia Microsystems and CDR CP/M and disk I/O interface cards.

Featuring: Complete installation instructions • Mounting bracket • 90 day warranty

Field reliability record exceeding 3 years

Only $65.00 Shipping & Handling $5.00



2 PORT SERIAL — CARD — I/O 3 PORT PARALLEL

". . . not your typical vanilla-flavored serial and parallel interface. . ."

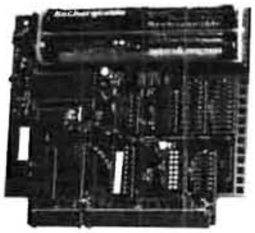
Your H/Z 88, 89, 90 can now directly connect and operate EPSON, IDS, ANADEX, GEMINI, SILVER REED, NEC, SUPER 5, PROWRITER, OKIDATA, and many more line printers using CENTRONICS style parallel interface with our **2/3rds**, 2 port serial, 3 port parallel interface card. Or you may use all 24 digital lines in various configurations of input, output, or bidirectional modes for industrial control or data sampling

Features:
• 2 Serial Ports Supporting Ring Input, and External Clock • 3 Parallel Ports, 24 Total Digital Input/Output Lines • Fully Compatible with All Models of H/Z 88, 89, 90 using Heath/Zenith CP/M or HDOS • Now Supporting CP/M Version 2 2 04 • Choice of Centronics Line Printer Support Software for the CP/M or HDOS Operating Systems • Reduced Computer Bus Loading and Chip Independent Design

Complete with Installation instructions, Documentation, 90 Day Warranty, Two Serial Cables and a Parallel Cable Internal to tne Computer

Price $199.00 **Second Operating System Driver $25.00**
Shipping & Handling $10.00

REAL TIME CLOCK



You will be able to perform time and date stamping for point of sales software, and bulletin board software or perform time studies as well as real time data sampling with our **REAL TIME CLOCK**. This peripheral card is a perfect companion to our 2/3rds card for industrial control and data sampling. STOP WATCH time study and alarm demonstration software is included for either the CP/M or HDOS operating systems. You will be able to view the current date and time on screen continuously or simply listen to an audible beep every fifteen minutes and the hour chimed or disable the clock entirely at your option

Features:
• True I/O Addressing, Not Memory Mapped • User Selectable Address • Rechargeable Battery Backup Using Commonly Available Batteries • Installable on the Left or Right Side of the Computer (Left side operation requires our I/O expansion module) • Month, Day, Year, Hours, Minutes, Seconds, 1/10's, 1/100's and 1/1000's of Second Accuracy • Interrupt Capability Based on Tenths of Seconds, Seconds, Minutes, Hour, Day, Week or a Specific Date and Time • Choice of CP/M or HDOS Operating System Software Driver and Demonstration Programs

Price $105.00 with Batteries **$89.00 without Batteries**
$ 25.00 Software for Second Operating System
Shipping & Handling $5.00

HDOS is a reg. trademark of the Heath Co CP/M is a reg. trademark of Digital Research
PRICES ARE LESS SHIPPING AND TAX IF RESIDENT OF CALIFORNIA
MAIL ORDER: 12011 ACLARE ST., CERRITOS, CA 90701 (213) 924-6741
TECHNICAL INFO/HELP 8575 KNOTT AVE., SUITE D, BUENA PARK, CA 90620 (714) 952-3930

ZENITH data systems
SERVICE CENTER

TERMS AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE — VISA AND MASTER CARD GLADLY ACCEPTED

RESOLUTION, SPEED, AND SILENCE

Superior Graphics and A Pseudo Disk for $495.

That's right, silence! Because when you plug the Interactive Graphics Controller/ Pseudo Disk into your H/Z89 or H/Z19-H8 system you not only get graphics that rival most 16 bit computers, you also get a high speed Pseudo Disk that does the work of another disk drive in your system, only much faster, and without all of that noise!

This upgrade stretches the horizons of your system into the future with High Resolution Graphics, Pseudo Disk Storage, 2 Trackball/ Joystick Interfaces, and 2 Parallel Printer Ports, all for one very competitive price.

Copy your CP/M or HDOS files to the Pseudo Disk and cut execution times as much as 80% over mechanical drives with disk I/O intensive programs. Files are even retained when you cold boot, and HDOS users can sysgen it as the boot device!

Take a good look at these features: • 3 pages of graphics, 640 by 250 pixels each • Mix text and graphics with independent scrolling • A remarkably easy-to-use high level driver that gives ANY language the power of fast graphics • Unmatched drawing speeds result from an advanced design which requires no handshaking or wait states • Use interlace mode and double the screen resolution to 640 by 500 pixels (requires Super19 or Ultra ROM) • Plug in up to 2 trackballs or joysticks for interactive applications (provisions for light pen) • 2 Centronics parallel printer ports with full support available • An alternate character set ROM socket to double the number of characters your screen can display (Script character set available) • Source codes available • European 50Hz compatible • Mounts professionally and easily into your H/Z89 or H/Z19-H8 and no existing features are sacrificed • 90 day limited warranty.

COMPARE	Northwest Digital Graphics-Plus	Cleveland Codonics Imaginator	SigmaSoft and Systems
Normal Pixel Resolution	512 × 250	504 × 247	640 × 250
Interlaced Pixel Resolution	n/a	n/a	640 × 500
Graphics Memory Size	16k	16k	64k/128k*
Number of Graphics Pages	1	1	3
Pseudo Disk Memory Size	n/a	n/a	64k/128k*/256k*
Super19/Ultra ROM Compatible	No	No	Yes
Vertical Graphics Scrolling	No	No	Yes
Type of Interface	Serial	Serial	Parallel
Independent Power Supply	No	No	Yes
Graphics Input Device Ports	n/a	n/a	2
Parallel Printer Ports	1*	n/a	2
Price/Performance Ratio	Good	Better	Best

*Option at additional charge.

Call or write NOW so we can rush our complete information package to you!

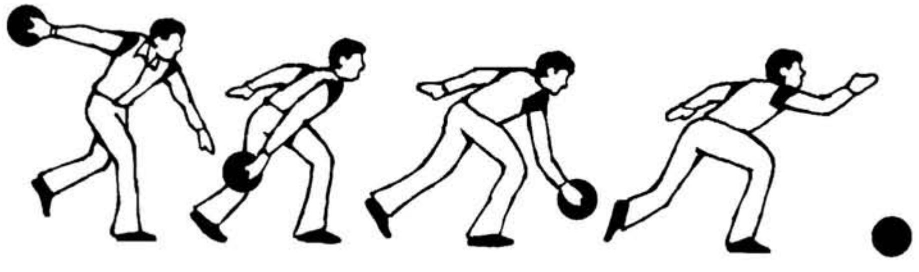
SigmaSoft
AND SYSTEMS

"Support is the most important feature."

4488 Spring Valley #107, Dallas, TX 75234, (214) 392-1025

David E. Warnick
RD#2 Box 2484
Spring Grove, PA. 17362

Sequential Files



Part 1

This article begins our discussions of, and work with sequential files. This method of file handling, just like the random files we discussed in past months, has specific uses which make it favorable in certain applications. We'll look at these applications, then create, sort, and use sequential files. The information you get here will let you decide whether you should use random or sequential files for your particular application. We'll even write programs to convert from sequential to random and from random to sequential files to give you the maximum power you need to use your data efficiently.

This series will consist of five articles. In your correspondence about my past work, many of you indicated a distaste for all the typing which must be done to enter programs and run them. So, I've created a disk with all the programs that will be necessary for this series. It's available for CP/M only. The programs are written in MBASIC, version 5.21. I'll reproduce the disk in either hard- or soft-sectored format on my H-89. All formats are the standard Heath version. If you would like a copy of this disk, drop me a line, ask for disk #4, specify hard or soft sectoring, and include $5.00 to cover the cost of the disk and shipping. My address appears at the top of this article.

These articles are copyrighted by the author. You are welcome to use this information and the knowledge you may gain from it for any purpose you desire, but you may not copy or sell any portion of it without the written permission of HUG and the author. Someday, I'll get around to writing a book, I hope, and I intend to use some of this information in it.

As you read these articles and write and run the programs presented here, you may have questions, or ideas for a better way of doing things. Please write to the address at the beginning of this article and I'll be happy to help. Keep your questions in line with the subject matter presented here, and include a stamped self-addressed envelope for the reply if one is necessary.

That's enough time spent on the preliminaries. Let's get on with our new world of sequential files.

With sequential files, all the information stored in them is packed on the disk or tape. No blank space is left. This makes for very efficient use of the storage media, no matter what it may be. However, when we want to find anything in the file, we must start at the beginning of the file and read everything until we get to what we want. To change a file, we've got to rewrite it as a temporary file. To sort sequential files, all of their contents must be read into memory and handled there.

So why would anybody use sequential rather than random access files, and why should we learn about them? First, there are more sequential files than random files in existence. You're bound to run into them sooner or later, so you should understand what's going on. With tape drives and cassette operation, only sequential files will work. You cannot jump all around on a tape, looking for what you want, the way you can with a disk. It's important to be able to convert from sequential to random, and from random to sequential files in order that you will be able to best use your data. To do this, you must understand both methods of storing data. Finally, any application which requires that all entries (records) in a file be read (as in printing labels for a mailing list) is done more efficiently by the disk operating system if sequential files are used. If a disk sector is read, it is used completely. There's no jumping around all over the disk to read the file entries in order.

Let's look at how information is stored in sequential files. We'll begin with a review of the terminology we use. A complete assemblage of information is called a FILE. We give it a name and store it on our disk or tape. Within a file, we find several repetitions of similar information. This could be name, address, and phone number for several people. Each of these groups of data is called a RECORD. Each record gives complete information about something. The data contained within a record has meaning by itself. Within these records there are smaller pieces of information which by themselves would be meaningless. Just a phone number from the above example wouldn't tell us anything without all the information from the record. These pieces of information are called FIELDS.

Let's look at how fields are placed in records, and how records are placed in files. You'll recall from our last series that random files allocated a specific number of characters to each field, and consequently to each record. Everything had a place to be, and we knew where it was because of that relationship. A random file might be set up as:

```
    Last Name        15 characters
    First Name       10 characters
    Middle Initial    1 character
```

and would look like

```
Warnick        David     E
↑   ↑    ↑    ↑    ↑    ↑↑
1   5    10   15   20   25 26
```

In a sequential file, no space is reserved. We're going to pack everything into the smallest space possible. We could do it like this

```
WarnickDavidE
```

but how would the computer know where one field ends and the next field begins? It's even worse with a complete entry like

```
WarnickDavidERD#2 Box2484SpringGrovePa17362
```

We'll have to put something into the file to show the limits of each field. We have a lot of characters to choose from, but we have 2 requirements to meet. First, the characters used should not have special meanings to the computer, the terminal, or the operating system (as in ASCII character 007 which rings the terminal bell). Second, we should choose a character which will not appear as a part of the information contained in any field of the information to be stored in the file. The backslash "\" and the tilde work very well in both cases. We're not likely to encounter either of these in the data we enter into our file. Whatever character we choose, we'll call it a DELIMITER, or more specifically in this case, a FIELD DELIMITER because it shows the limits of the fields within a record. Having made our choice, we could enter our record as:

```
Warnick\David\E\RD#2 Box2484\Spring Grove\Pa\17362
```

This way it'll be easy to separate the fields of the record. When we come to a backslash, we know we're entering the next field. You'll note as we progress that when there's no information for a given field (it'll probably be associated with some variable in our program) that we place a field delimiter in the file so the computer can keep track of where it is within the record. In the example above, if I had no middle initial, the file entry would be:

```
Warnick\David\\RD#2 Box2484\Spring Grove\Pa\17362
```

Without the extra delimiter, a program would read Warnick and assign it to Last Name, read David and assign it to First Name, then read RD#2 Box2484 and assign it to Middle Initial, etc. Therefore it will be important to preserve all fields, including those without any data, with field delimiters. You'll see this more clearly as we get into our programs.

It has probably occurred to you by now that if we needed to separate fields of the record with field delimiters, that we must separate the records within a file with RECORD DELIMITERS. We could use a second character, distinct from our field delimiter, or we could use a double field delimiter. (In our case, a double backslash.) That would look like:

```
\\Warnick\David\E\RD#2 Box 2484\
Spring Grove\Pa\17362\\Doe\
John\\Tree Street\Anytown\USA\99999\\Smith\
```

This is a very popular method, but one I wouldn't recommend. The example above, of a field without any information is the reason, and it's why I introduced it at this time. Does this mean that a triple backslash (\\\) is used, or that a different character should be used. Either would work, and the choice could be yours. All this discussion on record delimiters is academic, however, and I presented it to make you aware that they must be in the file. Fortunately for us, this is another case of the authors of MBASIC doing their homework and making our life easy. I discussed the record delimiter to make sure you understand that it is necessary and does exist. MBASIC inserts two characters, a carriage return and a line feed, into the file between records, and knows to stop at them when asked to read a record. Not all systems you may encounter in the future will do this for you. When you've created a sequential file later in this series you can see these otherwise invisible characters by using a utility such as ZAP or DDT and reading the HEX output of the file.

The result of this discussion is that you must choose your own field delimiters, but MBASIC will provide record delimiters for you.

When creating a sequential file our first task is to get the information we want for each field, insert the field delimiters, and let MBASIC insert the record delimiter. When we have accomplished this, we'll be able to open our file and put the information into it. For our examples, we'll let the screen scroll. Frozen and formatted screens were presented in past articles, so when you write your own application for a sequential file, you can see how to make it more user friendly in those back issues of REMark.

As we write our first program, we'll start at line 5000 to leave room for the opening of files which we'll add later. Our file will be a phone book with last name, first name, middle initial, and phone number. We'll get each piece of information in order and add delimiters as we go. We'll use the variable A$ for our input, and the variable R$ for the record we want to write into our file. Our program looks like this.

```
2 ' **** GETINFO.BAS
4 ' **** DAVID E. WARNICK
6 ' **** COPYRIGHT 1984
5000 'MAKE SURE THERE'S NOTHING IN R$
5010 R$=""
5020 'ASK FOR LAST NAME
5030 INPUT "LAST NAME ";A$
5040 'ADD LAST NAME TO RECORD
5050 R$=A$
5060 'ADD FIELD DELIMITER
5070 R$=R$+"\"
5080 'ASK FOR FIRST NAME
5090 INPUT "FIRST NAME ";A$
5100 'ADD FIRST NAME TO RECORD
5110 R$=R$+A$
5120 'ADD ANOTHER FIELD DELIMITER
5130 R$=R$+"\"
5140 'ASK FOR MIDDLE INITIAL
5150 INPUT "MIDDLE INITIAL ";A$
5160 'ADD MIDDLE INITIAL
     AND FIELD DELIMITER TO RECORD
5170 R$=R$+A$+"\"
5180 'ASK FOR PHONE NUMBER
5190 INPUT "PHONE NUMBER ";A$
5200 'ADD PHONE NUMBER TO RECORD
5210 R$=R$+A$
5220 'ALL DONE, PRINT THE RECORD
5230 PRINT "R$= ";R$
5240 'GO BACK AND DO IT AGAIN
5250 GOTO 5000
```

In our program, we first set R$ to nothing. This is done so that we can use a loop to enter several records. Lines 5030, 5090, 5150, and 5190 use the INPUT statement to prompt the operator for the information desired, and to assign that data to A$. Line 5050 puts the last name into R$, then line 5070 adds the first field delimiter. Later, lines 5110 and 5130 add the second field and another field delimiter to R$. This way R$ is appended (added to) with each new field and its delimiter.

In line 5170 we combined two steps to add a field and its delimiter to the record in one step. I added line 5230 to show you what you've created. This way, you can run our program as it is until we figure out how to add this data to the file.

As you can see, I've created an endless loop. When you run this program it will go on forever. To stop, you will have to hold down the CONTROL key (labeled CTRL) on your keyboard and type the letter C. This will return you to the MBASIC prompt.

Now type the program above, or copy it from Disk #4 if you ordered one, and run it. Enter names and phone numbers when they're asked for. Then take a close look at the record produced. Notice that you created a record, complete with field delimiters. Thus far, we haven't done anything with the record, and the next step will be to put it into a sequential file. As you play with this program, try leaving some (or all, for that matter) of the fields blank. Then look closely at the resulting record. Notice that the delimiters are there, but that there's nothing between them. The fields have been preserved, even though nothing may have been entered into them. If you type just spaces and a return, those spaces will be preserved as they are very real characters to the computer, so think about what you enter. Try the things you wonder about. You won't hurt anything and you may learn a lot. Who said learning can't be fun?

We'll use this program later in the series, so be sure to save it. Because we want to be able to combine the small programs we produce in various articles, save it in the ASCII format so the MERGE command will work. All the files on disk #4 are in the ASCII format so you won't need to modify them. Use the command:

```
SAVE "GETINFO",A
```

A lot of background into what sequential files are has been presented in this first article of the series, so we'll stop here to let you absorb it. Next month we'll begin by opening a sequential file and writing our phone book information to it. For those of you who can't wait, you can get ahead by looking up the commands used for sequential files in your MBASIC manual. (There he goes again with that "Read the manual" stuff.) They are:

```
OPEN
PRINT #
INPUT #
WRITE #
CLOSE
EOF()
LOC
```

That should keep you busy till the next fact-filled issue of REMark arrives. See you then.

HUG Price List

The following HUG Price List contains a list of all products not included in the HUG Software Catalog. For a detailed abstract of these products, refer to the issue of REMark specified.

HDOS HARDCOPY SOFTWARE

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1008	Volume I Documentation	9.00	
885-1013	Volume II Documentation	12.00	
885-1015	Volume III Documentation	9.00	
885-1037	Volume IV Documentation	12.00	8
885-1058	Volume V Documentation	12.00	

MISCELLANEOUS HDOS COLLECTIONS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1032	Disk V H8/89	18.00	8
885-1044-[37]	Disk VI H8/89	18.00	
885-1064-[37]	Disk IX H8/89 Disk	18.00	
885-1066-[37]	Disk X H8/89	18.00	10
885-1069	Disk XIII Misc H8/89	18.00	

GAMES

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1010	Adventure Disk H8/89	10.00	4
885-1029-[37]	Disk II Games 1 H8/89	18.00	8
885-1030-[37]	Disk III Games 2 H8/89	18.00	8
885-1031	Disk IV MUSIC H8 Only	20.00	25
885-1067-[37]	Disk XI H8/19/89 Games	18.00	12
885-1068	Disk XII MBASIC Graphic Games	18.00	10
885-1088-[37]	Disk XVII MBASIC Graphic Games	20.00	14
885-1093-[37]	D&D H8/89 Disk	20.00	16
885-1096-[37]	MBASIC Action Games H8/89	20.00	18
885-1103	Sea Battle HDOS H19/8/89	20.00	20
885-1111-[37]	HDOS MBASIC Games H8/89	20.00	23
885-1112-[37]	HDOS Graphic Games H8/89	20.00	23
885-1113-[37]	HDOS Action Games H8/89	20.00	23
885-1114	H8 Color Raiders & Goop	20.00	23
885-1124	HUGMAN & Movie Animation Pkg	20.00	41
885-1125	MAZEMADNESS	20.00	41
885-1130	Star Battle	20.00	47
885-1133-[37]	HDOS Games Collection I	20.00	59
885-8009-[37]	HDOS & CP/M Galactic Warrior	20.00	32
885-8022	HDOS SHAPES	16.00	45
885-8026	HDOS Space Drop	16.00	49
885-8032-[37]	HDOS Castle	20.00	59

CP/M

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1206-[37]	CP/M Games Disk	20.00	11
885-1209-[37]	CP/M MBASIC D&D	20.00	19
885-1211-[37]	CP/M Seabattle	20.00	20
885-1220-[37]	CP/M Action Games	20.00	32
885-1222-[37]	CP/M Adventure	10.00	35
885-1227-[37]	CP/M Cassino Gamess	20.00	38
885-1228-[37]	CP/M Fast Action Games	20.00	39
885-1236-[37]	CP/M Fun Disk I	20.00	55

ZDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-3004-37	ZDOS ZBASIC Graphic Games	20.00	37
885-3009-37	ZDOS ZBASIC D&D	20.00	50
885-3011-37	ZDOS ZBASIC Games Disk	20.00	52
885-3017-37	ZDOS Contest Games Disk	25.00	58

UTILITIES

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1022-[37]	HUG Editor (ED) Disk H8/89	20.00	20
885-1025	Runoff Disk H8/89	35.00	
885-1060-[37]	Disk VII H8/89	18.00	
885-1061	TMI Load H8 ONLY Disk	18.00	
885-1062-[37]	Disk VIII H8/89 (2 Disks)	25.00	
885-1063	Floating Point Disk H8/89	18.00	
885-1065	Fix Point Package H8/89 Disk	18.00	10
885-1075	HDOS Support Package H8/89	60.00	
885-1077	TXTCON/BASCON H8/89	18.00	
885-1079-[37]	HDOS Page Editor	25.00	15
885-1080	EDITX H8/H19/H89 Disk	20.00	
885-1082	Programs for Printers H8/89	20.00	
885-1083-[37]	Disk XVI Misc H8/89	20.00	11
885-1089-[37]	Disk XVIII Misc H8/89	20.00	20
885-1090-[37]	Disk XIX Utilities H8/89	20.00	22
885-1092-[37]	Relocating Debug Tool H8/89	30.00	14
885-1098	H8 Color Graphics ASM	20.00	19
885-1099	H8 Color Graphics Tiny PASCAL	20.00	19
885-1105	HDOS Device Drivers H8/89	20.00	24
885-1116	HDOS Z80 Debugging Tool	20.00	27
885-1119-[37]	BHBASIC Support	20.00	29
885-1120-[37]	HDOS 'WHEW' Utilities	20.00	33
885-1121	HDOS Hard Sec Sup Pkg 2 disks	30.00	37
885-1123	XMET Robot & Cross Assembler	20.00	40
885-1126	HDOS Utilities by PS:	20.00	42
885-1127-[37]	HDOS Soft Sector Support Pkg	30.00	45
885-1128-[37]	HDOS DISKVIEW	16.00	46
885-1129-[37]	HDOS CVT Color Video Terminal	20.00	46
885-8001	SE (Screen Editor)	25.00	28
885-8003	BHTOMB	25.00	28
885-8004	UDUMP	35.00	28
885-8006	HDOS SUBMIT	20.00	31
885-8007	EZITRANS.	30.00	30
885-8015	HDOS TEXTSET Formatter	30.00	42
885-8017	HDOS Programmers Helper	16.00	42
885-8024	HDOS BHBASIC Utilities Disk	16.00	46

CP/M

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1210-[37]	CP/M ED (same as 885-1022)	20.00	20
885-1212-[37]	CP/M Utilities H8/89	20.00	21
885-1213-[37]	CP/M Disk Utilities H8/89	20.00	22
885-1217-[37]	HUG Disk Duplication Utilities	20.00	26
885-1223-[37	]HRUN HDOS Emulator 3 disks	40.00	37
885-1225-[37]	CP/M Disk Dump & Edit Utility	30.00	40
885-1226-[37]	CP/M Utilities by PS:	20.00	40
885-1229-[37]	XMET Robot & Cross Assembler	20.00	40
885-1230-[37]	CP/M Function Key Mapper	20.00	42
885-1231-[37]	Cross Ref Utilities for MBASIC	20.00	43
885-1232-[37]	CP/M Color Video Terminal	20.00	46
885-1235-37	CP/M COPYDOS	20.00	54
885-1237-[37]	CP/M Utilities	20.00	55
885-5001-37	CP/M 86 KEYMAP	20.00	51
885-5002-37	CP/M 86 HUG Editor	20.00	52
885-5003-37	CP/M 86 Utilities by PS:	20.00	54
885-8018-[37]	CP/M FAST EDDY & BIG EDDY	20.00	43
885-8019-[37]	DOCUMAT and DOCULIST	20.00	43
885-8025-37	CP/M 85/86 FAST EDDY	20.00	49

ZDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-3005-37	ZDOS ETCHDUMP	20.00	39
885-3007-37	ZDOS CP/EMulator	20.00	47
885-3008-37	ZDOS Utilities	20.00	47
885-3010-37	ZDOS KEYMAP	20.00	51
885-8029-37	ZDOS FAST EDDY	20.00	53

H/Z100 ZDOS - H/Z150 MSDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-3012-37‡	ZDOS HUG Editor	20.00	52
885-3014-37‡	ZDOS/MSDOS Utilities II	20.00	54
885-3016-37†	ZDOS/MSDOS Adventure	10.00	57

† All program files run on both
‡ Program files run partially on both

PC/IBM COMPATIBLE

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-6001-37	MSDOS Keymapper	20.00	59
885-6002-37	CP/EMulator II & ZEMulator	20.00	59

PROGRAMMING LANGUAGES

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1038-[37]	Wise on Disk H8/89	18.00	
885-1042-[37]	PILOT on Disk H8/89	19.00	
885-1059	FOCAL-8 H8/89 DISK	25.00	13
885-1078-[37]	HDOS Z80 Assembler	25.00	21
885-1085	PILOT Documentation	9.00	
885-1086-[37]	Tiny HDOS Pascal H8/89	20.00	13
885-1094	HDOS Fig-Forth H8/89 2 Disks	40.00	18
885-1132-[37]	HDOS Tiny BASIC Compiler	20.00	59

CPM

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1208-[37]	CP/M Fig-Forth H8/89 2 Disks	40.00	18
885-1215-[37]	CP/M BASIC-E	20.00	26

BUSINESS, FINANCE AND EDUCATION

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1047	Stocks H8/89 Disk	18.00	
885-1048	Personal Account H8/89 Disk	18.00	
885-1049	Income Tax Records H8/89 Disk	18.00	
885-1055-[37]	MBASIC Inventory Disk H8/89	30.00	
885-1056	MBASIC Mail List	30.00	
885-1070	Disk XIV Home Fin H8/89	18.00	
885-1071-[37]	MBASIC SmBusPk H8/H19/H89	75.00	17
885-1091-[37]	Grade/Score Keeping H8/89	30.00	14
885-1097-[37]	MBASIC Quiz Disk H8/89	20.00	18
885-1118-[37]	MBASIC Payroll	60.00	30
885-1131-[37]	HDOS CHEAPCALC	20.00	47
885-8010	HDOS CHECKOFF	25.00	32
885-8021	HDOS Student's Statistics Pkg	20.00	44
885-8027	HDOS SCICALC	20.00	50

CP/M

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1218-[37]	CP/M MBASIC Payroll	60.00	31
885-1233-[37]	CP/M CHEAPCALC	20.00	47
885-1239-[37]	Spread Sht. Contest Disk I	20.00	
885-1240-[37]	Spread Sht. Contest Disk II	20.00	
885-1241-[37]	Spread Sht. Contest Disk III	20.00	
885-1242-[37]	Spread Sht. Contest Disk IV	20.00	
885-1243-[37]	Spread Sht. Contest Disk V	20.00	
885-1244-[37]	Spread Sht. Contest Disk VI	20.00	
885-8011-[37]	CP/M CHECKOFF	25.00	32

ZDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-3006-37	ZDOS CHEAPCALC	20.00	47
885-3013-37	ZDOS Checkbook Manager	20.00	54
885-3018-37	ZDOS Contest Spreadsheet Disk	25.00	58
885-8028-37	ZDOS SCICALC	20.00	50
885-8030-37	ZDOS MATHFLASH	20.00	55

DATA BASE MANAGEMENT SYSTEMS

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1107-[37]	HDOS Data Base System H8/89	30.00	23
885-1108-[37]	HDOS MBASIC Data Base Sys.	30.00	23
885-1109-[37]	HDOS Retriever ASM (3 disks)	40.00	23
885-1110	HDOS Autofile (2 disks)	30.00	23
885-1115-[37]	HDOS Navigational Program	20.00	25
885-8008	Farm Accounting System	45.00	30

CP/M

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1219-[37]	CP/M Navigational Program	20.00	31

AMATEUR RADIO

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-8016	Morse Code Transceiver Ver 2.0	20.00	42

CP/M

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1214-[37]	CP/M MBASIC Log Book (64k)	30.00	23
885-1234-[37]	CP/M Ham Help	20.00	49
885-1238-[37]	CP/M ASCRITY	20.00	57
885-8020-[37]	CP/M RF Comp. Aided Design	30.00	44
885-8031-[37]	CP/M Morse Code Transceiver	20.00	57

COMMUNICATION

HDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1122-[37]	HDOS MicroNET Connection	16.00	37

CP/M

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-1207-[37]	CP/M TERM & HTOC	20.00	26
885-1224-[37]	CP/M MicroNET Connection	16.00	37
885-3003-]37]	CP/M ZTERM (Z100 Modem Pkg)	20.00	34
885-5004-37	CPM86 TERM86 and DSKED	20.00	56
885-5005-37	CP/M86 16 Bit Micronet Connect.	16.00	61
885-8005	MAPLE (Modem Appl. Effector)	35.00	29
885-8012-[37]	CP/M MAPLE (Modem Program)	35.00	34
885-8023-37	CP/M 85 MAPLE	35.00	45

ZDOS

Part Number	Decription of Product	Selling Price	Vol. - Issue
885-3019-37	ZDOS 16 Bit Micronet Connection	16.00	61

Vectored to Page 36 ☞

HUG NEW PRODUCTS

885-5006-37 CP/M-86
HUGPBBS $40.00

Introduction: HUGPBBS or HUG Personal Bulletin Board System is a program that turns your computer system into an electronic bulletin board service. This bulletin board can be closed or open to the general public. If this board is closed, each caller will be required to enter his own individual pre-assigned password, as well as his name before he can leave or kill messages, and upload or download files. In addition to this capability, it also has a program database from which a user can upload and download files. Another feature is its ability to allow a caller to communicate, by way of his keyboard and crt, with you, the Sysop, should he desire. Finally, one of the most powerful features it has is its ability to allow you to access your computer, at system level, from a remote location. This ability is protected by three levels of security and can be totally disabled at configuration time.

Requirements: HUGPBBS requires the Heath/Zenith CP/M-86 operating system on the H/Z-100, 101, 120, or 121 computer. A minimum of two disk drives, any size, are needed. One 5" drive is needed for the message base and one 5" drive for the program data base. If a larger drive is available for the program data base, a greater amount of files can be stored. The program database feature CANNOT be disabled to eliminate the need for two drives. Finally, an auto-answer type modem is required. This software was written around the Hayes Smartmodem, although it uses no Smartmodem English commands. All modem protocol is handled by way of the RS-232 signals. Aside from the standard signal and ground pins, pins 8 and 20 are used. Pin 8 is DCD, or data carrier detect. This line goes from -12v to +12v when a carrier is detected and vice versa when the carrier is lost. Pin 20 is DTR or data terminal ready. This line should be capable of causing the modem to disconnect when it goes to -12v. It should also prevent the modem from answering the phone when at this level. It is the responsibility of the user to make his modem work properly with this software if he is not using a Hayes Smartmodem. Three baud rates are supported by the software (110, 300, and 1200). It is not necessary, however, for the modem to be capable of all three. Although not necessary, a printer would allow the Sysop to see the name of each caller while he was away from the console, as well as get a hard copy of any message on the bulletin board. It would also enable any user to leave a short three line message to the Sysop upon exit from the board.

The following files are included on the HUG P/N 885-5006-37 HUGPBBS disk:

```
HUGPBBS  .CMD
SBULETIN .HPB
BULLETIN .HPB
NEWHELP  .HPB
USERLOG  .HPB
MERGE    .SUB
EDIT     .CMD
EDIT     .DOC
README   .DOC
```

An extensive users' manual is also included with this software.

Author: Jim Buszkiewicz (Heath Users' Group)

Program Content: The bulletin board portion of this software has the following commands available to the user when he calls in:

A - Auto Baud Rate Detect
B - Bulletins (file created by Sysop)
C - Catalog Of Program Data Base
D - Download A File
E - Enter A Message
G - Goodbye (disconnect)
H - Help
I - Information (file created by Sysop)
K - Kill (a message)
N - Name
R - Retrieve (a message)
SM - Scan And Match
SQ - Scan Quick
SR - Scan And Retrieve
SS - Scan Subject Headers
T - Talk (with Sysop)
U - Upload A file

When the board is inactive, the following commands are available to the Sysop at the main console:

↑C - Warm Boot (available anytime and disconnects)
↑E - Enter (message)
↑H - Help
↑K - Kill (ANY message)
↑Q - Quick Scan
↑R - Retrieve (message)
↑S - Scan
↑X - Configure System

Operating Notes: Uploading and downloading files to the program database can be achieved using either Xmodem protocol (with checksum error testing), or with a simple capture buffer (ascii files only). A standard 192k H/Z-100 system will have a 65520 byte capture buffer available for file uploading.

The Short Bulletin, Regular Bulletin, and Information Files can be created using any text editor or word processor. The standard CP/M-86 HUG Editor was included with this software for this purpose. This is the editor found on the HUG P/N 885-5002-37 disk. The Short Bulletin and Regular Bulletin files automatically appear when a user first signs on and can be recalled with the "B" command. The difference between the two bulletin files is that the Short Bulletin file is forced upon the user and must be read when he logs in. This file is usually kept quite short. The Regular Bulletin file is also presented to the user, but can be aborted at any time. The Information File will be displayed for the user only when the "I" command is issued.

The bulletin board system can contain a total of 128 messages. These messages can contain a maximum of 16 lines with 77 characters in each line. A simple line replacement editor is available to the user to edit a message he may be working on.

Messages can be protected from being killed by other users with a password of up to 5 characters. Both upper and lower case is recognized in this password (i.e. the password "FIDO" is not the same as "fido").

The "SM" (Scan and Match) command allows a user to search one of three fields (To, From, and Subject) to match a string which he enters. With this command a user could possibly search for any messages to him (from his girlfriend) or any subject matter. Character case is ignored in this command.

For a closed bulletin board system, user passwords must be issued and changed by you, the Sysop. A user cannot enter or change his own password.

Simplified software configuration was a key factor in the writing of this system and is accomplished by answering questions when the software is first executed. This configuration is then permanently recorded. Reconfiguring the system is just as easy and can be done at any time.

Some of the options you are allowed to change during configuration time are:

1. Drive designation where each system file is to be placed.
2. Open or closed bulletin board selection.
3. Dead time period (no keyboard activity from caller) to disconnect.
4. Enable system level access.

and more.

Files that are uploaded to the database are not immediately made available to the general public until the Sysop has had a chance to view the contents and validity of those files. Once these files are "merged" into the database by the Sysop, only then will the general public see them in the database catalog.

While a caller is using the bulletin board system, the Sysop is allowed to observe all activity on his CRT. During this period of time, the Sysop is not allowed to interfere with the activities of the caller other than to totally abort the program which immediately disconnects the phone connection.

Comments: Knowing how to use some sort of text editor or word processor is imperative to the successful operation of this software.

TABLE C Rating: (0), (1), (2), (5), (10)

885-5007-37 CP/M-86
HUGPBBS Source Listing $60.00

Description: This disk contains the source listing for the program "HUGPBBS," HUG P/N 885-5006-37. This listing contains all comments and can be assembled using Digital Researches' CP/M-86 assembler "ASM86." This source listing is being made available on an "as-is" basis and modifications made to it are done so at the sole responsibility of the user. No other files or documents are included with this product.

Requirements: An H/Z-100 system running CP/M-86 is required to assemble this program. A minimum of two 5" drives are also needed, since the source file is approximately 90k and the listing file generated will be between 250k and 300k.

The following file is included on the HUG P/N 885-5007-37 HUGPBBS Source Listing disk:

HUGPBBS .A86

Author: Jim Buszkiewicz (Heath Users' Group)

Comments: None

TABLE C Rating: n/a

TABLE C
Product Rating

10 - Very Good
9 - Good
8 - Average

Rating values 8-10 are based on the ease of use, the programming technique used, and the efficiency of the product.

7 - Has hardware limitations (memory, disk storage, etc.)
6 - Requires special programming technique
5 - Requires additional or special hardware
4 - Requires a printer
3 - Uses the Special Function Keys (f1,f2,f3,etc.)
2 - Program runs in *Real Time**
1 - Single-keystroke input
0 - Uses the H19 (H/Z89) escape codes (graphics, reverse video)

Real Time — a program that does not require interactivity with the user. This term usually refers to games that continue to execute with or without the input of the player, e.g. p/n 885-1103 or 885-1211[-37] SEA BATTLE.

ORDERING INFORMATION

For Visa and MasterCard phone orders; telephone Heath Company Parts Department at (616) 982-3571. Have the part number(s), descriptions, and quantity ready for quick processing. By mail; send order, plus 10% postage and handling ($1.00 minimum charge, up to a maximum of $5.00. UPS is $1.75 minimum -- no maximum on UPS. UPS Blue Label is $4.00 minimum.), to Heath Company Parts Department, Hilltop Road, St. Joseph, MI 49085. Visa and MasterCard require minimum $10.00 order.

Any questions or problems regarding HUG software or REMark magazine should be directed to HUG at (616) 982-3463. REMEMBER-Heath Company Parts Department is NOT capable of answering questions regarding software or REMark.

NOTE

The [-37] means the product is available in hard-sector or soft-sector. Remember, when ordering the soft-sectored format, you must include the "-37" after the part number; e.g. 885-1223-37.

885-8033-37 MS-DOS FAST EDIT Text Editor and BIG EDIT $20.00

Introduction: FAST EDIT is a text file screen editor that was written for everybody. It was written using the basic commands and keypad keys, so that anyone, even with no experience with an editor, can learn to use it while reading the instructions. For those files that are too large for your computer's memory, BIG EDIT will handle the breaking up of the text for editing with FAST EDIT.

Requirements: This disk requires the MS-DOS operating system on an H/Z-150/160 computer. A printer is not required, but both FAST EDIT and BIG EDIT have printer options. Only one disk drive is required. Since BIG EDIT can be used with large files, a second drive may be required to break up large files which cannot fit into memory. The original file is not changed or deleted.

The following files are included on the HUG P/N 885-8029-37 MS-DOS FAST EDIT Text Editor and BID EDIT disk:

FASTEDIT	.COM	TUTOR1	.DOC
INSTRUCT	.DOC	TUTOR2	.DOC
BIGED	.COM	TUTOR3	.DOC
BIGED	.DOC	README	.DOC
EDITAGN	.COM		

Author: Hubert L. Reeder

FAST EDIT - This text file screen editor and its documentation have been designed for anyone not familiar with using an editor. The program uses commands and keys that are easy to remember and use.

The editor contains a limited number of commands, however, the commands are designed to provide a useful, easy-to-use editor. It does not have complex options that require time and effort to use. The editor contains a command mode and edit mode. The following are a brief list of the options:

Command Mode

LOAD filename.ext	(load file)
SAVE filename.ext	(save file)
SAVE XX filename.ext	(save XX number of lines)
MERGE filename.ext	(merge two files)
PRINT	(print enter file, NN lines per page)
PRINT NN	(print double spaced)
FIND any word	(find first occurence of matched word)
MARGIN nn xx	(set left and right margins)
BYE	(exit to MS-DOS)

Key Commands

Up Arrow	enter EDIT mode at first line of text
Down Arrow	enter EDIT mode at last line of text
HOME	enter EDIT mode at pointer (last cursor location)
DELETE	cancel partial commands or stop printer

Edit Mode

Up Arrow	move cursor up one line
Down Arrow	move cursor down one line
Right Arrow	move cursor to the right one character
Left Arrow	move cursor to the left one character
HOME	return to command mode
+ (key)	insert line
- (key)	delete line
INS	insert character
DEL	delete character
PgUp	NN lines
PgDw	NN lines
F1	align paragraph
F2	justify right
F3	center line
F4	split line
F5	indent on/off
F6	margin on/off
F7	block erase
F8	help
F9	tab set/release
F10	find next word

These are most of the basic commands of FAST EDIT. Please note that it has the ability to align paragraphs to new margin settings and then the option of right justifying the paragraph text.

Details of how to use these options are contained in the documentation. The TUTOR1, TUTOR2, and TUTOR3 documentation files are included with the disk to give the user experience in using FAST EDIT while reading the doc files.

BIG EDIT - This program is a utility to work with text files which are too large to be edited by FAST EDIT directly, because of memory limitations. BIG EDIT can be used to browse a file of any size of which the user can break the large file into smaller parts for editing with FAST EDIT.

BIG EDIT asks for the input filename and an output filename. It keeps track of the subfiles and names them accordingly.

BIG EDIT has some useful options to aid the user in preparing the text for smaller files. The BROWSE mode is similar to the EDIT mode of FAST EDIT, except that no editing can be done to the file.

The following are some of the commands of BIG EDIT:

SAVEALL	save the entire text in memory to the disk
SAVEPART	save part of the text in memory to disk
NOSAVE	discard part of text
BYE	exit to MS-DOS

Comments: none

TABLE C Rating: (1), (3), (10)

Vectored to Page 64 ☞

☜ Vectored from Page 33

MISCELLANEOUS

885-0004	HUG Binder	5.75	
885-1221-[37]	Watzman ROM Source Code/Doc ...	30.00	33
885-4001	REMark Vol. I Issues 1-13	20.00	
885-4002	REMark Vol. II Issues 14-23	20.00	
885-4003	REMark Vol. III Issues 24-35	20.00	
885-4004	REMark Vol. IV Issues 36-47	20.00	
885-4005	REMark Vol. V Issues 48-59	25.00	
885-4500	HUG Software Catalog	9.75	
885-4600	Watzman/HUG ROM	45.00	41
885-4700	HUG Bulletin Board Handbook	5.00	50
885-3015-37	ZDOS SKYVIEWS	20.00	55

NOTE: The [-37] means the product is available in hard sector or soft sector. Remember, when ordering the soft sectored format, you must include the "-37" after the part number; e.g. 885-1223-37.

INTERROGATOR™

Drive Diagnostic Program for the ZENITH Z/H-100 Series

Diagnosing disc drive problems has just become a great deal easier.

Interrogator,™ Dysan's economical new drive diagnostics package, can run a drive through its pace in minutes. To tell you exactly what's wrong. Or right.

Just like that.

Test results come right up on the screen of the machine you're testing. Or you can print out hard copy records.

The Interrogator package includes a Digital Diagnostic Diskette,™ complete test instructions and software programmed to perform a whole battery of sophisticated tests.

Things like head radial alignment. Head positioner linearity. Head azimuth alignment. Index timing. Spindle speed. Read/write verification. And over a dozen others.

Interrogator is here now for the Zenith Z/H 100 Series. Testing both 5¼″ and 8″ models. Versions for other computer models are on the way.

Whether you're going for one drive or need to assure the compatibility of a company full of drives, Interrogator is the perfect tool.

And it's just the kind of thing you'd expect from Dysan. The leader in quality, innovative media and diagnostic tools.

Interrogator. What could be easier?

Just this. A toll free phone call to get more information. Or place an order. (800) 551-9000.

Dysan®

5201 Patrick Henry Drive
P.O. Box 58053
Santa Clara, CA 95050
(408) 988-3472

CALC.BAS

William E. Reese
1719 Middle Bellville Rd
Mansfield, OH 44904

Introduction

There it sat on my desk: $2000.00 (give or take a few dollars) worth of computer. I had installed a number of commercially available software products in it and had developed a number of BASIC applications programs to make my job easier, but it bothered me that in order to perform simple calculations, I still had to get out my handy HP-25 calculator. Of course, Microsoft does point out that MBASIC can be used as a calculator in the command mode, but this requires very un-calculator like entries (such as PRINT). Also, that method does not lend itself well to complex or repetitive calculations.

CALC.BAS is a full-featured calculator program written for CP/M version 2.2.03 and MBASIC version 5.21, which simulates most of the non-programmable functions of the Hewlett-Packard HP-25 calculator and features:

* 4 register operating stack
* 8 storage registers
* 4 basic arithmetic functions (+, -, X, /)
* advanced arithmetic functions (inversion, square, square root, general power, natural & common logs and anti-logs)
* trig functions with angles expressed in degrees, radians, or grads
* RPN (Reverse Polish Notation)

The program makes use of a number of features of the H/Z-89 or H-19 terminal including:

* function keys
* 25th line
* direct cursor addressing
* graphics
* reverse video

Other features are:

* help screens for all inputs
* error trapping, error messages and recovery

Program Description

Although the program listing is rather long, it is broken into a number of smaller routines for simplicity and to make it easy to customize it to your needs. For instance, it would be a simple matter to delete (or add) arithmetic functions, delete the help screens, or limit angular units to one or two choices instead of three. Except for lines 1, 2, and 3, all lines are consecutive multiples of 10, so you can use the AUTO function when typing in the listing. Also, all GOTO's and GOSUB's jump to an executable line so you may omit any or all REM's if you want to.

Lines 40-210 define variables, dimension an array, and enable error trapping. A few words about variable names: I prefer to use somewhat longer but more meaningful names. These make editing easier, especially if I haven't looked at the program for awhile.

Line 220 causes the program to power up with angles expressed in degrees. You could easily change it to power up in either of the other two units.

Line 230 clears the screen and turns the cursor off. I tried the program with the cursor on and with it off and prefer it off. You could easily leave it on if you prefer.

Lines 240 and 250 print the main screen and the initial function key labels. Note that the labels use two lines, the 24th and 25th.

Line 290 places the cursor in the X register.

Line 300 initializes the digit counter.

Line 310 inputs one byte at a time (either digit or function).

Line 320 limits the input to 20 digits.

Lines 330 and 340 allow input in exponential notation.

Line 350 checks to see if the input is a digit or function input.

Lines 360-390 raise the stack if the current input is the first digit of a new numeric input.

Line 400 properly positions a minus sign. The digits are input individually as strings but when entered, the number is printed as a numeric, so the leading space in a numeric representing the sign needs to be accounted for.

Line 420 prints each digit.

Line 430 increments the digit counter and returns to input the next digit.

Line 440 checks for a carriage return or ESCape. A carriage return will mean enter; ESCape will mean perform a function.

Line 450 jumps the "build X" routine if I=1, since there is no number to build.

Lines 490-520 build the value of X by successively concatenating all the entries in the A$(I) array and then taking the VALue of the result.

At line 530, the only possible values for the current A$(I) are carriage return or ESCape. If it is carriage return (ENTER), line 540 raises the stack if I=1 and line 550 prints the new value of X, and returns to line 290 for the next input. If it is ESCape, line 560 directs the program to the appropriate function key evaluation routine.

Lines 630-1620 evaluate function key inputs depending on which set of function key labels have been selected. Lines 630-760 are typical for ESC=1. If a function key has been struck, line 310 has input the byte representing ESCape using INPUT$(1) and the appropriate function key code byte is pending at the terminal. This byte is input in line 630 using INKEY$. The use of the INPUT$(1)/INKEY$ combination to input the function key code assures that only a function key input will be evaluated, since it is almost impossible to press ESCape and a letter key with the proper timing to simulate a function key.

Lines 640-730 evaluate the function key code and perform the appropriate function (arithmetic in lines 640-670; control in lines 680-730). Note that line 730 checks for no byte pending which corresponds to the ESCape key being struck rather than a function key. Lines 740-760 print the new stack values as required and return to line 290 for the next input. David Warnick (REMark #47 page 37) describes a faster, neater way of decoding function key inputs based on the ASCII value of the key code. I chose not to use this method, since CALC.BAS also makes use of the ERASE key (ASCII value 74) and the ESCape key (ASCII value 27). Since these ASCII values are not consecutive with the function keys, they would require additional decoding anyway.

The trig functions in lines 1090-1430 require some special attention. First, since MBASIC performs all angular calculations in radians, variable C is used to convert radians to degrees or grads depending on which units have been selected. Second, since a sine or cosine greater than 1 does not exist, line 1240 is used to generate a custom error if an arcsine or arccosine of a number greater than 1 is attempted. Third, the only intrinsic arc function in MBASIC is arctangent. Arcsine and arccosine are calculated using the formulas in lines 1250 and 1270 (page 7-11 in my MBASIC manual). As X approaches 1, SQR(-X*X+1) approaches 0, generating a "divide by 0" error message. Lines 1260, 1280, and 1320-1420 get around this problem by assigning values (rather than calculating them) to arcsine and arccosine if X gets too near 1. This method seems cumbersome, but it was the least cumbersome method I could come up with that worked; I would appreciate hearing from anyone who can come up with a better way of handling this problem.

CLEAR is also special since lines 1470-1510 first print new function key labels. A complete function key code is required to reach the CLEAR routine and another complete code is required to CLEAR anything (lines 1520 and 1540). Line 1550 simply moves the stack down to CLEAR X; line 1560 sets all stack values to 0; line 1570 sets all register values to 0; line 1580 sets both stack and register values to 0. Line 1610 prints the function key labels that existed before CLEAR was entered.

Like CLEAR, STORE and RECALL are reached using a complete function key code. In the case of STORE and RECALL, I have evaluated the ASCII value of the input and used ON/GOTO, since the ASCII values of the inputs are consecutive (decimal 0-7; ASCII 48-55). The only other allowable input is the RED function key (ESCape Q) which displays the help screen. At first glance, lines 1660-1680 (1820-1840 for recall) do not appear to be able to input a two byte function key code. However, if the RED key is pressed, INPUT$(1) in line 1660 (1820) inputs ESCape; line 1670 (1830) checks for "Q"; line 1680 (1840) checks for 0-7 and then returns to 1660 (1820) where "Q" is pending at the terminal. INPUT$(1) now inputs "Q" which is acted on in line 1670 (1830). This is the only case where an alternate key input will cause a function to be performed (shift Q will also display the help screen).

Lines 1690-1850 print the main screen including register and stack values.

Lines 1890-1940 print the stack values only using direct cursor addressing.

Lines 1980-2070 print the register values only.

Lines 2440-2520 determine which angular units will be used. These lines set the value of ANGLE which is used in line 1150 to determine which units are next, set the value of ANGLE$ which is printed as the label for the BLUE key, and set the value of C which is used to convert radians to the selected units (C=1 when radians are selected). When the units are changed, the complete function key labels in screen lines 24 and 25 are reprinted, but since only the BLUE key label changes, the BLUE key label appears to toggle from DEG to RAD to GRD and back to DEG.

The END routine in lines 2560-26010 is not a true subroutine since obviously there is no RETURN, but I didn't know where else to put it. To reach this routine, you must have pressed the ESCape key. Lines 2560 and 2570 look for your next input which must at least start with ESCape (either the ESCape key or a function key). If it is does not, the END routine is aborted and you return to line 290 for another input. If the input is ESCape, lines 2580 and 2590 assure that it was the ESCape key rather than a function key; if not, you again return to line 290. This sequence assures that you END only when you really want to. Line 2600 returns the terminal to "normal" and line 2610 ENDS. One obvious modification is to replace END in line 2610 with some other function; I used CHAIN "MENU" in my version to return me to my master program menu.

Function key labels are printed in lines 2680-3050. Variable ESC is used to remember which set of labels is in use when CLEAR is selected (line 1610). Each set of labels uses reverse video and block graphics to print on screen lines 24 and 25. To help you count blanks between "i's": each total label is preceeded and followed by a blank (i.e. " + "). Two line labels have extra blanks to account for characters in the other line. For instance to print the label for X squared (lines 2800 and 2820), the 24th line contains the leading blank, a blank for X in the line below, the 2, and the trailing blank. The 25th line contains the leading blank, the X, a blank for 2 in the line above, and the trailing blank. Note that when labels containing lower case letters are printed (line 2920) graphics mode must be switched on and off repeatedly, since lower case letters printed with graphics on produce graphics characters instead of the desired letters.

Help screen texts are contained in lines 3120-4330. Line 3120 is typical of what happens when a help screen is accessed. When a help screen is selected, the cursor is in the X register. The PRINT drops the cursor one line so it is below all the printing on the screen (except the function key labels). ERAUP$ erases to the beginning of the screen (everything but the function key labels). The cursor is positioned at the upper left corner and the help screen is printed. The screen for ESC=1 is actually a two page screen, the first giving general information, the second explaining the function key labels. The screens for all labels except STORE and RECALL are exited by pressing any key and you are returned to where you were before you selected the help screen. To exit the STORE or RECALL help screen, you must press a number key (0-7) to select a register to store in or recall from.

Lines 4370-4510 contain the error trapping routine. Various error messages are defined depending on which error occurs (including custom error 255 in line 4400), and the message is printed in the X register. When you press any key as directed in line 4470, the message is erased and the last value of X is restored. After considerable testing, no errors not covered in this routine have occurred, however if they do, the routine can easily be changed to accommodate them.

Reverse Polish Notation

Wher using RPN, you ENTER your first number, key your second number, press the desired function key, and the result is displayed; there is no equal key. For instance, to add 43 to 57, key 43, press ENTER (or RETURN), key 57, and press f1 (+). The answer 100 will be displayed.

The first time user may find this method strange, but a little use will demonstrate its usefulness. It is especially handy when doing things like multiple sums of products (A*B)+(C*D)+(E*F)+... or when solving problems with lots of nested parenthesis.

Program Operation

Load and run CALC.BAS. The main screen will appear along with the initial set of function key labels. To add two numbers (43 and 57 as in the example above), key 4 then 3; 43 will be displayed in the X register. Press ENTER or RETURN; 43 is still displayed in X. Key 5 then 7. As soon as you press 5, 43 will move to the Y register and 5 then 57 will appear in X. Press f1 (+); the answer to 43+57 (100) will appear in the X register and 43 and 57 will disappear. To use this result in a further calculation (multiply by 76, for instance) simply key 7 then 6. When you press 7, the 100 in the X register will move to the Y register and 7 then 76 will appear in X. Press f3 (X) and the result of 100 X 76 (7600) will be displayed in X. Follow this same procedure for the other basic arithmetic functions.

To store the value of X in a storage register, press f5 (STOR) followed by a number from 0 to 7 to select the desired register; the value of X will appear in the selected register, as well as in X.

To recall a number from a storage register to the X register, press BLUE (RCL) followed by a number from 0 to 7 to select the desired register; the value in the selected register will appear in X and the stack will raise.

To clear an entry, press ERASE (CLR). A new set of function key labels will appear giving you the choice of CLEARing the X register, the memory stack (registers X, Y, Z, and T), the storage registers (R0-R7), or everything. You will also be given the choice of aborting the CLEAR function by again pressing ERASE (now labeled ABORT).

Press RED (?) at any time to display the appropriate help screen.

Press WHITE (NEXT) to select the next set of function key labels.

Most of the other functions (1/X, X squared, ln, SIN, etc.) do not require that the number be entered first. Simply key in the number and press the desired function key.

When using the trig functions, the label of the BLUE key will indicate the angular units currently selected. To select different units, press the BLUE key repeatedly until the desired units are displayed. To display the sine, cosine, or tangent of X, press the appropriate function key. To display the inverse or arc function, press f5 (ARC) followed by the desired function; f1 (SIN) for arcsine, for instance.

To end the program at any time, press the ESCape key twice.

Additional Modifications

If you CHAIN in and out of a set of programs repeatedly and often as I do, you may find the time required to load and run CALC.BAS to be objectionable. There are two things you can do to speed it up. One has already been mentioned: delete all REM's. The second is to delete, or at least remove from the main program, all the help screen texts. If you are the only one that will use the program and you fully understand it, the simple thing to do is delete this function entirely. If, as in my case, others who are not as familiar with it may use CALC.BAS or if you are just a purest who doesn't like an incomplete program, you can create the help screens as ASCII text files and access them only when needed. This is what I have done. The changes required in the program listing to accomplish this are shown in Listing 2.

Finally, if you don't need all the fancy functions of CALC.BAS and/or are a proponent of the KISS principal, take a look at "A Spiffy Little Program" by R. C. Perkins (Buggin' HUG REMark #47 page 50).

```
1 REM                   CALC.BAS
2 REM              by William E. Reese
3 REM
10 REM ****************************************
20 REM *******          Initialize          *******
30 REM ****************************************
40 DIM A$(20)
50 DEFSNG T:DEFSNG X-Z:DEFSNG R:DEFINT B:DEFINT I-J
60 ON ERROR GOTO 4370
70 E$=CHR$(27)
80 CLR$=E$+CHR$(69)
90 CUROFF$=E$+"x5"
100 CURON$=E$+"y5"
110 DEF FNDIR$(L,C)=E$+"Y"+CHR$(L+31)+CHR$(C+31)
120 E25OFF$=E$+"y1"
130 E25$=E$+"x1"+E$+"Y"+CHR$(56)+CHR$(32)
140 ERA$=E$+"K"
150 ERAUP$=E$+"b"
160 REV$=E$+"p"
170 NOR$=E$+"q"
180 GRAFOFF$=E$+"G"
190 GRAFON$=E$+"F"
200 ESC=1
210 PI=3.14159
220 ANGLE$=" DEG ":ANGLE=1:C=2*PI/360
230 PRINT CLR$;CUROFF$
240 GOSUB 2020: REM Print screen
250 GOSUB 2680: REM Print function key labels
260 REM ************************************
270 REM *****         Input routine         *****
280 REM ************************************
290 PRINT FNDIR$(19,13);
300 I=1
310 A$(I)=INPUT$(1)
320 IF I=20 THEN 440
330 IF A$(I)="e" THEN A$(I)="E"
340 IF A$(I)="E" THEN 360
350 IF ASC(A$(I))<45 OR ASC(A$(I))>57 OR ASC(A$(I))=47
    THEN 440
360 IF I=1 THEN 370 ELSE 420:
    REM Raise stack on first digit
370 T=Z:Z=Y:Y=X
380 GOSUB 2220
390 PRINT FNDIR$(19,13);ERA$;
400 IF A$(I)="-" THEN PRINT A$(I);
    ELSE PRINT " ";A$(I);
410 GOTO 430
420 PRINT A$(I);:REM Print each digit
430 I=I+1:GOTO 310
440 IF A$(I)=CHR$(13) OR A$(I)=CHR$(27)
    THEN 450 ELSE 310
450 IF I=1 THEN 530
460 REM
470 REM ******        Build the value of X        ******
480 REM
490 FOR J=1 TO I-1
500 A$=A$+A$(J)
510 NEXT J
520 X=VAL(A$)
530 IF A$(I)=CHR$(27) THEN 560
540 IF I=1 THEN T=Z:Z=Y:Y=X
550 GOSUB 2220:GOTO 290:
    REM Print new values of X, Y, Z, & T
560 ON ESC GOTO 630,800,960,1090:
    REM Evaluate function key input
570 REM ********************************************
580 REM *******    Function key evaluations    *******
590 REM ********************************************
610 REM **********            ESC=1            **********
620 REM
630 X$=INKEY$
640 IF X$="S" THEN X=X+Y:GOTO 750:      REM     + add
650 IF X$="T" THEN X=Y-X:GOTO 750:      REM - subtract
660 IF X$="U" THEN X=Y*X:GOTO 750:      REM X multiply
670 IF X$="V" THEN X=Y/X:GOTO 750:      REM   / divide
680 IF X$="W" THEN 1660:                REM      Store
690 IF X$="P" THEN 1820:                REM     Recall
700 IF X$="J" THEN 1470:                REM      Clear
710 IF X$="Q" THEN 3120:                REM       Help
720 IF X$="R" THEN GOSUB 2780:GOTO 290: REM       Next
730 IF X$="" THEN 2560:                 REM       Exit
740 GOTO 760
750 Y=Z:Z=T:T=0
760 GOSUB 2220:GOTO 290
770 REM
780 REM **********            ESC=2            **********
790 REM
800 X$=INKEY$
810 IF X$="S" THEN X=1/X:GOTO 920:      REM      Invert
820 IF X$="T" THEN X=X↑2:GOTO 920:      REM      Square
830 IF X$="U" THEN X=X↑.5:GOTO 920:     REM Square root
840 IF X$="V" THEN X=Y↑X:GOTO 910:      REM       Power
850 IF X$="W" THEN SWAP X,Y:GOTO 920:   REM Interchange
860 IF X$="J" THEN 1470:                REM       Clear
870 IF X$="Q" THEN 3450:                REM        Help
880 IF X$="R" THEN GOSUB 2880:GOTO 290: REM        Next
890 IF X$="" THEN 2560:                 REM        Exit
900 GOTO 920
910 Y=Z:Z=T:T=0
920 GOSUB 2220:GOTO 290
930 REM
940 REM **********            ESC=3            **********
950 REM
960 X$=INKEY$
970 IF X$="S" THEN X=LOG(X):GOTO 1050:REM     Natural log
980 IF X$="T" THEN X=EXP(X):GOTO 1050:REM Natural antilog
990 IF X$="U" THEN X=(LOG(X))/(LOG(10)):
    GOTO 1050:                         REM     Common log
1000 IF X$="V" THEN X=10↑X:GOTO 1050: REM  Common antilog
1010 IF X$="J" THEN 1470:             REM           Clear
1020 IF X$="Q" THEN 3640:             REM            Help
1030 IF X$="R" THEN GOSUB 2980:GOTO 290:REM          Next
1040 IF X$="" THEN 2560
1050 GOSUB 2220:GOTO 290
1060 REM
1070 REM **********            ESC=4            **********
1080 REM
1090 X$=INKEY$
1100 IF X$="S" THEN X=SIN(X*C):GOTO 1430:REM      Sine
1110 IF X$="T" THEN X=COS(X*C):GOTO 1430:REM    Cosine
1120 IF X$="U" THEN X=TAN(X*C):GOTO 1430:REM   Tangent
1130 IF X$="W" THEN 1200:               REM Arc functions
1140 IF X$="J" THEN 1470:                  REM     Clear
1150 IF X$="P" THEN ON ANGLE GOSUB 2480,2520,2440:GOTO 290
1160 IF X$="Q" THEN 3800:                  REM      Help
1170 IF X$="R" THEN GOSUB 2680:GOTO 290:  REM      Next
1180 IF X$="" THEN 2560:                   REM      Exit
1190 GOTO 290
1200 X$=INPUT$(1)
1210 IF X$<>CHR$(27) THEN 1200
1220 X$=INKEY$
1230 IF X$="U" THEN X=ATN(X)/C:GOTO 1430
```

```
1240 IF ABS(X)>1 THEN ERROR 255
1250 IF X$="S" AND ABS(X)<.99999
     THEN X=ATN(X/SQR(-X*X+1))/C:GOTO 1430
1260 IF X$="S" AND ABS(X)>.99999
     THEN ON ANGLE GOTO 1320,1340,1360
1270 IF X$="T" AND ABS(X)<.99999
     THEN X=(-ATN(X/SQR(-X*X+1))+1.5708)/C:GOTO 1430
1280 IF X$="T" AND ABS(X)>.99999
     THEN ON ANGLE GOTO 1380,1400,1420
1290 IF X$="Q" THEN 3800
1300 IF X$="" THEN 2560
1310 GOTO 1200
1320 IF X>0 THEN X=90 ELSE X=-90
1330 GOTO 1430
1340 IF X>0 THEN X=1.5708 ELSE X=-1.5708
1350 GOTO 1430
1360 IF X>0 THEN X=100 ELSE X=-100
1370 GOTO 1430
1380 IF X>0 THEN X=0 ELSE X=180
1390 GOTO 1430
1400 IF X>0 THEN X=0 ELSE X=PI
1410 GOTO 1430
1420 IF X>0 THEN X=0 ELSE X=200
1430 GOSUB 2220:GOTO 290
1440 REM
1450 REM **********            Clear            **********
1460 REM
1470 PRINT FNDIR$(23,1);REV$;GRAFON$
1480 PRINT "iiiiii   ii         i         i     iiiiii
     iiiiiii   iiiiiiiiiiiiii";
1490 PRINT E25$
1500 PRINT "iiiiii X ii STACK i REG'S i ALL iiiiii
     ABORT iiiiiii ? iiiiiiiiiiiiii";
1510 PRINT GRAFOFF$;NOR$;FNDIR$(19,13);
1520 X$=INPUT$(1)
1530 IF X$<>CHR$(27) THEN 1520
1540 X$=INKEY$
1550 IF X$="S" THEN X=Y:Y=Z:Z=T:T=0:GOSUB 2220:
     GOTO 1610
1560 IF X$="T" THEN X=0:Y=0:Z=0:T=0:GOSUB 2220:
     GOTO 1610
1570 IF X$="U" THEN R0=0:R1=0:R2=0:R3=0:R4=0:R5=0:R6=0:
     R7=0:GOSUB 2310:GOTO 1610
1580 IF X$="V" THEN R0=0:R1=0:R2=0:R3=0:R4=0:R5=0:R6=0:
     R7=0:X=0:Y=0:Z=0:T=0:GOSUB 2220:GOSUB 2310:GOTO 1610
1590 IF X$="J" THEN 1610
1600 IF X$="Q" THEN 3970 ELSE 1520
1610 ON ESC GOSUB 2680,2780,2880,2980
1620 GOTO 290
1630 REM
1640 REM **********           Store            **********
1650 REM
1660 X$=INPUT$(1)
1670 IF X$="Q" THEN 4130
1680 IF ASC(X$)<48 OR ASC(X$)>55 THEN 1660
1690 B=ASC(X$)-47
1700 ON B GOTO 1710,1720,1730,1740,1750,1760,1770,1780
1710 R0=X:GOSUB 2310:GOTO 290
1720 R1=X:GOSUB 2310:GOTO 290
1730 R2=X:GOSUB 2310:GOTO 290
1740 R3=X:GOSUB 2310:GOTO 290
1750 R4=X:GOSUB 2310:GOTO 290
1760 R5=X:GOSUB 2310:GOTO 290
1770 R6=X:GOSUB 2310:GOTO 290
1780 R7=X:GOSUB 2310:GOTO 290
1790 REM
1800 REM **********          Recall           **********
1810 REM
1820 X$=INPUT$(1)
1830 IF X$="Q" THEN 4250
1840 IF ASC(X$)<48 OR ASC(X$)>55 THEN 1820
1850 B=ASC(X$)-47
1860 T=Z:Z=Y:Y=X
1870 ON B GOTO 1880,1890,1900,1910,1920,1930,1940,1950
1880 X=R0:GOSUB 2220:GOTO 290
1890 X=R1:GOSUB 2220:GOTO 290
1900 X=R2:GOSUB 2220:GOTO 290
1910 X=R3:GOSUB 2220:GOTO 290
1920 X=R4:GOSUB 2220:GOTO 290
1930 X=R5:GOSUB 2220:GOTO 290
1940 X=R6:GOSUB 2220:GOTO 290
1950 X=R7:GOSUB 2220:GOTO 290
1960 REM ********************************************
1970 REM **********       Subroutines        **********
1980 REM ********************************************
1990 REM
2000 REM **********    Print main screen     **********
2010 REM
2020 PRINT TAB(8);REV$;" Storage Registers ";NOR$;
     TAB(49);REV$;"  ?   for help     ";NOR$
2030 PRINT TAB(8);REV$;"     ";NOR$;TAB(49);REV$;
     " ESC twice to exit ";NOR$
2040 PRINT TAB(8);REV$;" R0 ";NOR$;" ";R0
2050 PRINT TAB(8);REV$;" R1 ";NOR$;" ";R1
2060 PRINT TAB(8);REV$;" R2 ";NOR$;" ";R2
2070 PRINT TAB(8);REV$;" R3 ";NOR$;" ";R3
2080 PRINT TAB(8);REV$;" R4 ";NOR$;" ";R4
2090 PRINT TAB(8);REV$;" R5 ";NOR$;" ";R5
2100 PRINT TAB(8);REV$;" R6 ";NOR$;" ";R6
2110 PRINT TAB(8);REV$;" R7 ";NOR$;" ";R7
2120 PRINT:PRINT:PRINT TAB(8);REV$;" Memory stack ";NOR$
2130 PRINT TAB(8);REV$;"   ";NOR$
2140 PRINT TAB(8);REV$;" T ";NOR$;"  ";T
2150 PRINT TAB(8);REV$;" Z ";NOR$;"  ";Z
2160 PRINT TAB(8);REV$;" Y ";NOR$;"  ";Y
2170 PRINT TAB(8);REV$;" X ";NOR$;"  ";X
2180 RETURN
2190 REM
2200 REM *********    Print stack values     *********
2210 REM
2220 PRINT FNDIR$(16,13);ERA$;T
2230 PRINT FNDIR$(17,13);ERA$;Z
2240 PRINT FNDIR$(18,13);ERA$;Y
2250 PRINT FNDIR$(19,13);ERA$;X
2260 A$=""
2270 RETURN
2280 REM
2290 REM **********    Print register values  *********
2300 REM
2310 PRINT FNDIR$(4,13);ERA$;R0
2320 PRINT FNDIR$(5,13);ERA$;R1
2330 PRINT FNDIR$(6,13);ERA$;R2
2340 PRINT FNDIR$(7,13);ERA$;R3
2350 PRINT FNDIR$(8,13);ERA$;R4
2360 PRINT FNDIR$(9,13);ERA$;R5
2370 PRINT FNDIR$(10,13);ERA$;R6
2380 PRINT FNDIR$(11,13);ERA$;R7
2390 A$=""
2400 RETURN
2410 REM
2420 REM **********          Degrees          *********
2430 REM
2440 ANGLE=1:ANGLE$=" DEG ":C=2*PI/360:GOSUB 2980:RETURN
2450 REM
2460 REM **********          Radians          *********
2470 REM
2480 ANGLE=2:ANGLE$=" RAD ":C=1:GOSUB 2980:RETURN
2490 REM
2500 REM **********           Grads           *********
2510 REM
2520 ANGLE=3:ANGLE$=" GRD ":C=2*PI/400:GOSUB 2980:RETURN
2530 REM
2540 REM **********            End            *********
2550 REM
2560 X$=INPUT$(1)
2570 IF X$<>CHR$(27) THEN 290
2580 X$=INKEY$
2590 IF X$<>"" THEN 290
2600 PRINT CLR$;GRAFOFF$;NOR$;E25OFF$;CURON$
2610 END
2620 REM ********************************************
2630 REM **********   Define function keys   **********
2640 REM ********************************************
2650 REM
2660 REM **********          ESC=1            **********
2670 REM
```

```
2680 ESC=1
2690 PRINT FNDIR$(23,1);REV$;GRAFON$
2700 PRINT "iiiiii   iiii   iiii   iii   iii      ii
     ii     ii   iii      iiiiii";
2710 PRINT E25$;
2720 PRINT "iiiiii + iiii - iiii X iii / iii STOR ii CLR
     ii RCL ii ? iii NEXT iiiiii";
2730 PRINT GRAFOFF$;NOR$
2740 RETURN
2750 REM
2760 REM **********              ESC=2            **********
2770 REM
2780 ESC=2
2790 PRINT FNDIR$(23,1);REV$;GRAFON$
2800 PRINT "iiii     iii  2 iii  .5 i  X iii X->Y ii
     iiiiiiiii   iii      iiiiii";
2810 PRINT E25$;
2820 PRINT "iiii 1/X iii X  iii X   i Y  iii X<-Y ii CLR
     iiiiiiiii ? iii NEXT iiiiii";
2830 PRINT GRAFOFF$;NOR$
2840 RETURN
2850 REM
2860 REM **********              ESC=3            **********
2870 REM
2880 ESC=3
2890 PRINT FNDIR$(23,1);REV$;GRAFON$
2900 PRINT "iii        ii  X i          i   X iiiiiiiii
     iiiiiiiii   iii      iiiiii";
2910 PRINT E25$
2920 PRINT "iii";GRAFOFF$;" ln(X) ";GRAFON$;"ii";GRAFOFF$;
     " e  ";GRAFON$;"i";GRAFOFF$;" log(X) ";GRAFON$;
     "i 10  iiiiiiiii CLR iiiiiiiii ? iii NEXT iiiiii";
2930 PRINT GRAFOFF$;NOR$
2940 RETURN
2950 REM
2960 REM **********              ESC=4            **********
2970 REM
2980 ESC=4
2990 PRINT GRAFON$;REV$
3000 PRINT FNDIR$(23,1)
3010 PRINT "iiiii    ii     ii     iiiiiiii     iii     ii
     ii   iii      iiiiii";
3020 PRINT E25$;
3030 PRINT
     "iiiii SIN ii COS ii TAN iiiiiiii ARC iii CLR ii";
     ANGLE$;"ii ? iii NEXT iiiiii";
3040 PRINT GRAFOFF$;NOR$
3050 RETURN
3060 REM ********************************************
3070 REM *******         Help screens         *******
3080 REM ********************************************
3090 REM
3100 REM **********              ESC=1            **********
3110 REM
3120 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3130 PRINT "CALC is a calculator simulator which allows
     you to enter numbers"
3140 PRINT "using a parenthesis-free, unambiguous method
     called RPN (Reverse"
3150 PRINT "Polish Notation).":PRINT
3160 PRINT "To perform an operation involving 2 numbers
     (e.g. addition),"
3170 PRINT "key in the first number, press ENTER
     (or RETURN), key in the"
3180 PRINT "second number, then press the appropriate
     function key (f1 for"
3190 PRINT "+ in this case).":PRINT
3200 PRINT "To perform an operation involving 1 number,
     simply press the"
3210 PRINT "appropriate function key."
3220 PRINT:PRINT "Press any key to continue."
3230 Y$=INPUT$(1):Y$=INKEY$
3240 PRINT ERAUP$;FNDIR$(1,1)
3250 PRINT TAB(10);"+  addition (adds the contents of
     the X and Y registers)"
3260 PRINT TAB(10);"-  subtraction (subtracts the
     contents of the X register"
3270 PRINT TAB(13);"from the Y register)"
3280 PRINT TAB(10);"X  multiplication (multiplies the
     contents of the X and"
3290 PRINT TAB(13);"Y registers"
3300 PRINT TAB(10);"/ division (divides the contents of
     the X register into"
3310 PRINT TAB(13);"the Y register)"
3320 PRINT TAB(7);"STOR places the contents of the X
     register in the specified"
3330 PRINT TAB(13);"storage register"
3340 PRINT TAB(8);
     "CLR  clear (clears the selected register(s))"
3350 PRINT TAB(8);"RCL recall (places the contents of the
     specified storage"
3360 PRINT TAB(13);"register into the X register)"
3370 PRINT TAB(10);"?  help (displays these instructions)"
3380 PRINT TAB(7);"NEXT changes the function menu (total
     of 4 are available)"
3390 PRINT:PRINT TAB(13);"Press any key to continue."
3400 Y$=INPUT$(1)
3410 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3420 REM
3430 REM **********              ESC=2            **********
3440 REM
3450 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3460 PRINT TAB(8);"1/X  inversion (inverts X)"
3470 PRINT TAB(10);"2"
3480 PRINT TAB(9);"X   square (squares X)"
3490 PRINT TAB(9);".5"
3500 PRINT TAB(8);
     "X square root (takes the square root of X)"
3510 PRINT TAB(10);"X"
3520 PRINT TAB(9);"Y   power (raises Y to the X power)"
3530 PRINT TAB(7);"X->Y"
3540 PRINT TAB(7);"X<-Y swap (exchanges the values in
     the X and Y registers)"
3550 PRINT TAB(8);
     "CLR  clear (clears the selected register(s))"
3560 PRINT TAB(10);"? help (displays these instructions)"
3570 PRINT TAB(7);"NEXT changes the function menu
     (a total of 4 are available)"
3580 PRINT:PRINT TAB(13);"Press any key to continue."
3590 Y$=INPUT$(1)
3600 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3610 REM
3620 REM **********              ESC=3            **********
3630 REM
3640 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3650 PRINT TAB(6);
     "ln(X) natural log (returns the natural log of X)"
3660 PRINT TAB(10);"X"
3670 PRINT TAB(9);
     "e   natural antilog (raises e to the X power)"
3680 PRINT TAB(5);
     "log(X)  common log (returns the common log of X)"
3690 PRINT TAB(10);"X"
3700 PRINT TAB(8);
     "10   common antilog (raises 10 to the X power)"
3710 PRINT TAB(8);
     "CLR  clear (clears the selected register(s))"
3720 PRINT TAB(10);"? help (displays these instruction)"
3730 PRINT TAB(7);"NEXT changes the function menu
     (a total of 4 are available)"
3740 PRINT:PRINT TAB(13);"Press any key to continue."
3750 Y$=INPUT$(1)
3760 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3770 REM
3780 REM **********              ESC=4            **********
3790 REM
3800 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3810 PRINT "These are the trig functions.
     The program will accept angles"
3820 PRINT "expressed in DEGREES, RADIANS, or GRADS.
     When the program is"
3830 PRINT "loaded, it is set for degrees.
     Pressing the DEG key changes"
3840 PRINT "the units to radians and causes the DEG
     key to change to RAD."
```

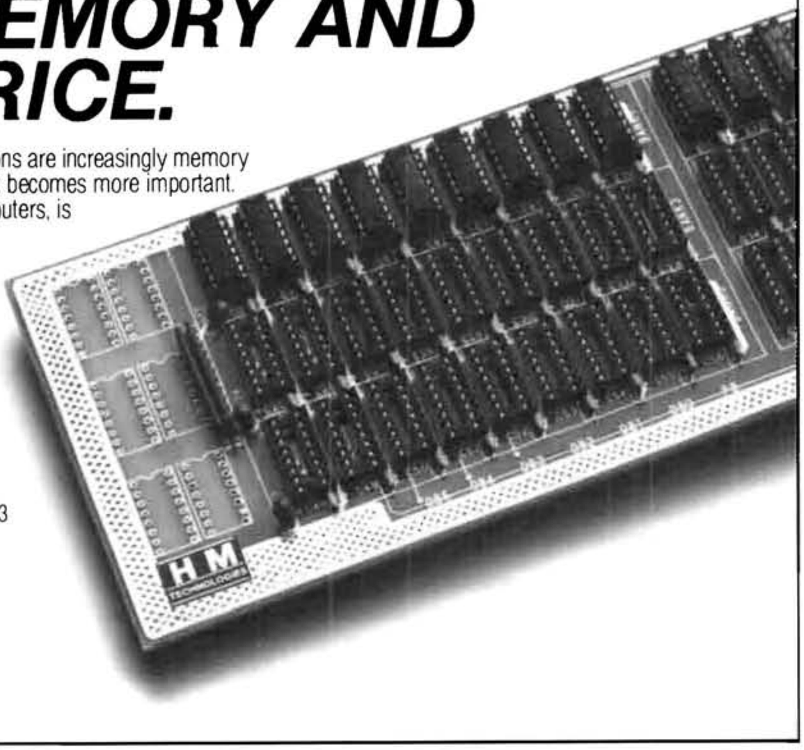
```
3850 PRINT "Pressing it again causes the units to
     change to grads and the"
3860 PRINT "key lable to change to GRD.":PRINT
3870 PRINT "To perform the function,
     simply press the appropriate key"
3880 PRINT "(e.g. SIN). To perform the inverse function,
     first press the"
3890 PRINT "ARC key followed by the function key
     (e.g. ARC followed by"
3900 PRINT "SIN yields arcsine)."
3910 PRINT:PRINT "Press any key to continue."
3920 Y$=INPUT$(1)
3930 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3940 REM
3950 REM **********          Clear          **********
3960 REM
3970 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3980 PRINT "You have pressed the CLR (clear) key either
     intentionally or"
3990 PRINT "accidentally, and the program is waiting
     for you to tell it"
4000 PRINT "what you want to clear.
     You have 4 choices:":PRINT
4010 PRINT TAB(10);"X  only the X register"
4020 PRINT TAB(6);
     "STACK the memory stack (X, Y, Z, and T registers)"
4030 PRINT TAB(6);"REG'S  the storage registers"
4040 PRINT TAB(8);"ALL  all registers":PRINT
4050 PRINT "If you do not want to clear anything,
     press the ABORT key."
4060 PRINT:PRINT "Press any key to continue."
4070 Y$=INPUT$(1)
4080 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020
4090 PRINT FNDIR$(19,13);:GOTO 1520
4100 REM
4110 REM **********          Store          **********
4120 REM
4130 PRINT:PRINT ERAUP$;FNDIR$(1,1)
4140 PRINT:PRINT:PRINT
4150 PRINT "You have pressed the STOR (store) key either
     intentionally or"
4160 PRINT "accidentally, and the program is waiting
     for you to tell it"
4170 PRINT "which Storage Register you want to use.
     Press any number from"
4180 PRINT "0 to 7 inclusive."
4190 X$=INPUT$(1)
4200 IF ASC(X$)<48 OR ASC(X$)>55 THEN 4190
4210 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 1690
4220 REM
4230 REM **********          Recall          **********
4240 REM
4250 PRINT:PRINT ERAUP$;FNDIR$(1,1)
4260 PRINT:PRINT:PRINT
4270 PRINT "You have pressed the RCL (recall) key either
     intentionally or"
4280 PRINT "accidentally, and the program is waiting for
     you to tell it"
4290 PRINT "which Storage Register you want to use.
     Press any number from"
4300 PRINT "0 to 7 inclusive."
4310 X$=INPUT$(1)
4320 IF ASC(X$)<48 OR ASC(X$)>55 THEN 4310
4330 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 1850
4340 REM *********************************************
4350 REM *******          Error trapping          *******
4360 REM *********************************************
4370 IF ERR=5 THEN 4410
4380 IF ERR=6 THEN MESSAGE$="That number is too large."
     :GOTO 4460
4390 IF ERR=11 THEN MESSAGE$=
     "Division by 0 is not allowed.":GOTO 4460
4400 IF ERR=255 THEN MESSAGE$=
     "Arcsine or arccosine of a number >1 is not allowed."
     :GOTO 4460
4410 IF ERL=830 THEN MESSAGE$=
     "The square root of a negative number is not allowed."
     :GOTO 4460
4420 IF ERL=840 THEN MESSAGE$=
     "Y negative and X not an integer is not allowed."
     :GOTO 4460
4430 IF ERL=970 THEN MESSAGE$=
     "Ln of 0 or a negative number is not allowed."
     :GOTO 4460
4440 IF ERL=990 THEN MESSAGE$=
     "Log of 0 or a negative number is not allowed."
     :GOTO 4460
4450 ON ER½OR GOTO 0
4460 PRINT FNFIR$(19,12);ERA$;MESSAGE$
4470 PRINT FNDIR$(20,12);"Press any key to continue."
4480 X$=INPUT$(1)
4490 PRINT FNDIR$(19,12);ERA$;FNDIR$(20,12);ERA$
4500 GOSUB 2220
4510 RESUME 290
```

```
3060 REM *********************************************
3070 REM *******          Help screens          *******
3080 REM *********************************************
3090 REM
3100 REM *******            ESC=1              *******
3110 REM
3120 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3130 FILENAME$="HELP1"
3140 GOSUB 4331
3230 Y$=INPUT$(1):Y$=INKEY$
3240 PRINT ERAUP$:FNDIR$(1,1)
3250 FILENAME$="HELP2"
3260 GOSUB 4331
3400 Y$=INPUT$(1)
3410 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3420 REM
3430 REM *******            ESC=2              *******
3440 REM
3450 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3460 FILENAME$="HELP3"
3470 GOSUB 4331
3590 Y$=INPUT$(1)
3600 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3610 REM
3620 REM *******            ESC=3              *******
3630 REM
3640 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3650 FILENAME$="HELP4"
3660 GOSUB 4331
3750 Y$=INPUT$(1)
3760 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3770 REM
3780 REM *******            ESC=4              *******
3790 REM
3800 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3810 FILENAME$="HELP5"
3820 GOSUB 4331
3920 Y$=INPUT$(1)
3930 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 290
3940 REM
3950 REM *******            Clear              *******
3960 REM
3970 PRINT:PRINT ERAUP$;FNDIR$(1,1)
3980 FILENAME$="HELP6"
3990 GOSUB 4331
4070 Y$=INPUT$(1)
4080 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020
4090 PRINT FNDIR$(19,13);:GOTO 1520
4100 REM
4110 REM *******            Store              *******
4120 REM
4130 PRINT:PRINT ERAUP$;FNDIR$(1,1)
4140 FILENAME$="HELP7"
4150 GOSUB 4331
4190 X$=INPUT$(1)
```

```
4200 IF ASC(X$)<48 OR ASC(X$)>55 THEN 4190
4210 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 1690
4220 REM
4230 REM *******                Recall              *******
4240 REM
4250 PRINT:PRINT ERAUP$;FNDIR$(1,1)
4260 FILENAME$="HELP8"
4270 GOSUB 4331
4310 X$=INPUT$(1)
4320 IF ASC(X$)<48 OR ASC(X$)>55 THEN 4310
4330 PRINT ERAUP$;FNDIR$(1,1):GOSUB 2020:GOTO 1850
4331 OPEN "I",#1,FILENAME$
4332 FOR I=1 TO 25
4333 LINE INPUT#1,A$
4334 IF EOF(1) THEN 4337
4335 PRINT A$
4336 NEXT I
4337 CLOSE#1
4338 RETURN
```

About the Author

Bill **Reese** *has his BEE degree from Ohio State University and is Manager of Engineering for Peabody Barnes, Inc., where he has implemented a data acquisition system in their test lab based on an H-89. He has built three H-89's, one for himself and two for work. His hobbies include electronics, computers (obviously), ham radio (my call is WB8ZRN), photography, and reading.*



The Ultimate In Dot Matrix?

(A review of the Printek 930 printer)

Jim Buszkiewicz
HUG Software Developer

Oh no, not another dot matrix printer review article! Yup, but this is not just another dot matrix printer. Read on and find out why.

Shortly after Zenith purchased Heath Company in 1980, a few Heath engineers formed a small local company called Printek Inc. It was their decision to design and market a dot matrix printer capable not only of high speed draft printing, but also capable of very high executive (letter) quality print. Today, Printek employs roughly 100 people and markets the model 930 printer, which is capable of the original design specifications, as well as being fully dot addressable.

The first task I had upon receiving this unit, was to construct a cable to go between the printer and my H-110. The printer comes standard with a Centronics parallel type interface, as well as an RS-232 serial type interface. Unfortunately, no interconnect cables of any sort are supplied. In fact, they must be purchased from some other source, or personally constructed. Since it was easiest for me to obtain parts for a serial cable, I decided to use the RS-232 interface instead of the parallel one. The manual clearly defines the function of each used pin on both the serial, as well as the parallel interface connectors. The following diagram illustrates the cable connections needed to interface this printer to either an H/Z-100 or H/Z-150 system:

H/Z-100 Cable Connections (serial)

H/Z-100 J1	Printek Serial Connector
1	1
3	3
4	11
7	7

H/Z-150/160 Cable Connections (serial)

H/Z-150/160 P-503	Printek Serial Connector
1	1
2	3
5	11
7	7

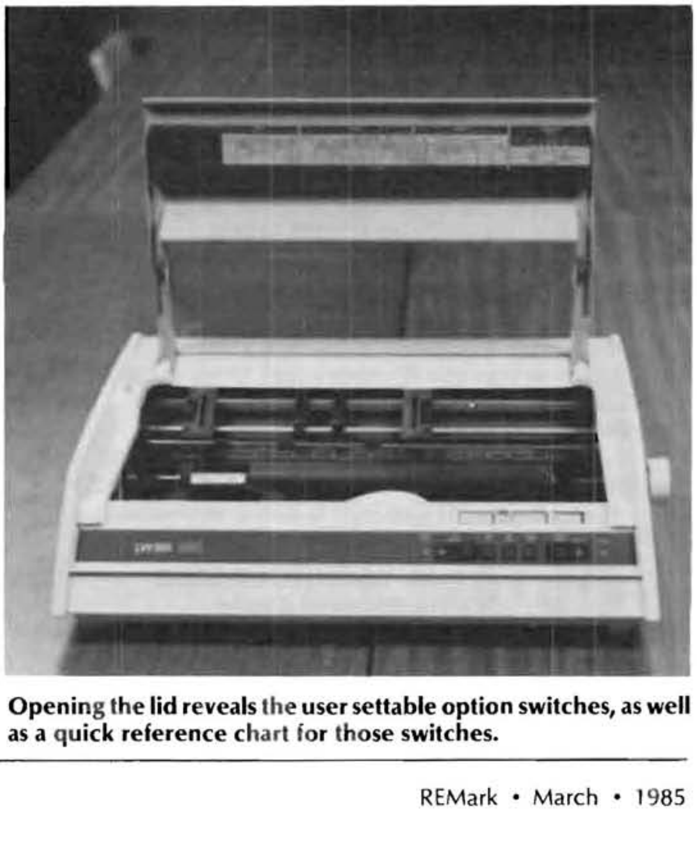
Since I did not have the proper parallel type connectors available to me, I could not verify proper operation of the parallel interface. According to the Printek manual, however, the parallel cable can be made up for the IBM PC on a 1-to-1 correspondence between the pins, within the connectors, at each end of the interface cable. This should also work with the H/Z-150/160.

The printer busy signal is available on pin 11 of the RS-232 connector. This signal is applied to pin 4 (pin 5 on the PC) or RTS input on the computer's RS-232 connector. The polarity of this busy signal is selectable by way of an internal switch setting inside the printer. Once the setting of this switch has been determined, you will then need to configure the printer port using the CONFIGUR program. This will be true for both computers and operating systems.

Most of the printer's operating characteristics are switch selectable. Some of the switchable functions are: font select, text select, language select, character pitch, proportional spacing, auto line feed, lines per inch, form length, baud rate, parity, handshake type, and busy polarity. From these switches I was able to select the initial operating characteristics of the printer.

My first interest was simply to see it print in both draft and executive quality modes. This was accomplished by using the control-P function of CP/M-86. Draft mode gives you the type of print you would expect from a dot matrix printer. Executive Quality mode produces print you expect from a daisy wheel printer! You can switch between both modes whenever the printer is inactive from one of the front panel switches.

The print produced in the executive quality mode was so good, I was more than curious to know how Printek did it. It appears that most of



Opening the lid reveals the user settable option switches, as well as a quick reference chart for those switches.

the magic is in the printhead construction. According to the manual, the printhead in the 930 has two staggered rows of nine, 12 mil. diameter wires, on .00694" centers. In Draft mode, a 9 x 9 dot matrix is used, and in Executive Quality mode a 36 x 18 dot matrix. There are two graphic densities, 72 x 72 dots per inch, and 144 x 144 dots per inch. This higher resolution mode gives you a whopping 20,736 dots per square inch! The following text is an example of the Draft and Executive Quality print produced by the 930:

This is an example of 10 characters per inch Executive Quality printing.
ABCDEFGHIJKLMNOPQRSTUVWXYZ

This is an example of 10 characters per inch Draft Mode printing.
ABCDEFGHIJKLMNOPQRSTUVWXYZ

As a programmer, my next interest was how fast can this printer produce a hardcopy of an assembly language source listing. It just so happened that I've been working on a program called HUGPBBS for several months and the source listing had been presently tallying out around 270k! That to me sounded like a good test for average throughput. I set up the printer for 16.7 characters per inch, initialized the computer, and hit the go button on my stop watch. About 40 to 50 minutes later (I don't remember exactly now), the printer had finished its task. A few moments later, some simple calculations revealed an average throughput of 94 characters per second.

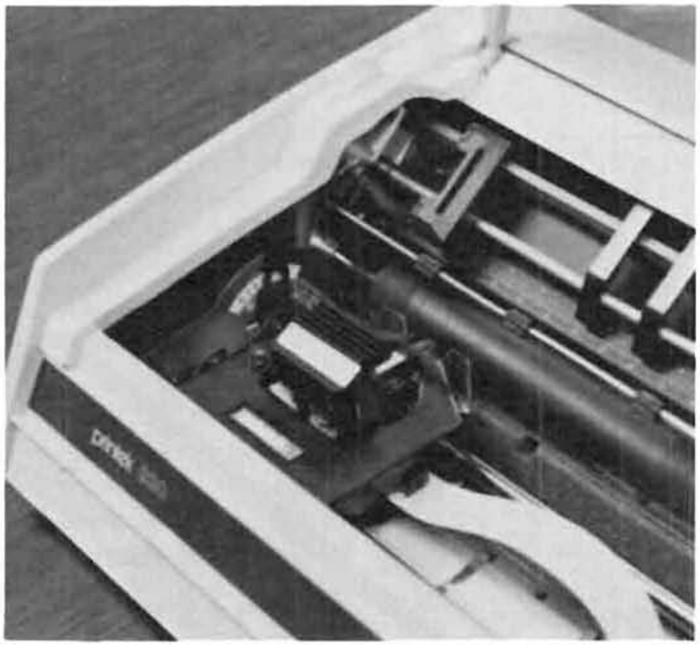
One of the biggest advantages this printer has over others I've used is that the Printek has both a tractor feed and pinch wheel paper feed mechanism. This feature comes in very handy when you need to frequently switch between fanfold paper and single sheets as often I do.

Up until now I haven't mentioned too much about the Graphics mode. Unfortunately, Printek does not supply any software drivers for any of the Heath/Zenith systems. Reproducing screen graphics requires such driver software and would have to be written by the user. The manual does thoroughly explain how the graphics can be accessed and the following example was included:

This is a box □ within a line

For those of you that need automatic sheet feeders, Printek has two models available: the ASF-1 single bin and the ASF-2 dual bin feeders. Both units have Diablo compatible sheet feeder commands and sell for $699 and $995, respectively.

A close-up view of the user settable option switches and the front panel switches.



A view of the head and paper feed mechanisms. Note the head-to-platten spacing adjustment lever.

With the features that come standard, this printer is beyond a doubt the machine for the professional programming or business environment.

Performance Features At A Glance:

Modes:	Executive Quality Draft Graphics
Draft Mode Print Speed:	200 cps at 10 cpi Bidirectional/Logic Seeking
Executive Quality Print Speed:	80 cps at 10 cpi Single Pass
Graphics Mode Print Speed:	4608 dots/second standard Bidirectional/Logic Seeking

Character Sets: 96 US ASCII character plus 25 additional characters to provide seven foreign languages. Character spacing of 10, 12, 13.3, and 16.7 characters per inch. Double width characters, proportional spacing, and coincident underlining.

Line Length: 132 columns at 10 cpi to 220 columns at 16.7 cpi. 1958 dots in 144 x 144 dots per inch graphics, and 979 dots in 72 x 72 dpi graphics.

Line Spacing: 6 and 8 lines per inch and programmable in 1/48 inch increments, plus 72 and 144 lpi in graphics.

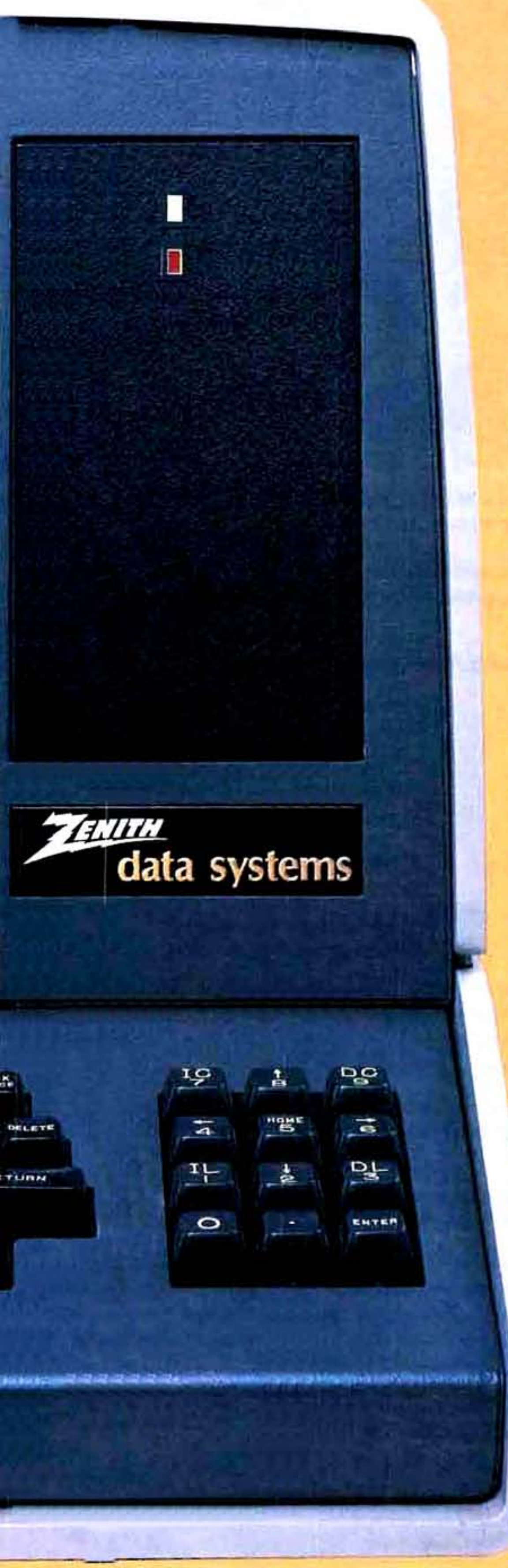
Paper Handling: Single sheets and continuous tractor feed, 1 to 6 parts. Roller platen friction feed for cut sheet, adjustable tractors for fanfold from 2.5 inch to 16 inch widths. Stepper motor drive with a slew rate of 5 inches per second. Top, bottom, front and rear loading.

Interface: Industry compatible RS-232-C serial and parallel are standard. 768 character buffer expandable by 14k characters. Diablo compatible software control.

Power Requirements: 120/240 vac +10% / -15% @ 47 to 63 hz.

Price: $1995.00

GREAT DRIVES!



The H89 TWOET systems Just Got Hard to beat!

We've got a great idea for your H-88, 89 or 90. It's a dual internal half height drive system. Two of our half height 5¼" drives can replace your built-in disk drive, doubling your information storage capacity.

Floppy Disk Services provides you with everything you need. That's two double-sided, double density, half height drives in either 48 or 96 tpi format, all hardware, cables and power connector adaptors. And most important, you get easy, step-by-step instructions, in the Heath/Zenith tradition of good, clear documentation.

We've thoroughly tested the TWOET/Heath set-up. And now by popular demand, the new 10 megabyte hard disk system! We are now able to offer you the storage you want and need for a fair price! The hard disk system comes with all software needed to run with the 17 or 37 controller and/or the Magnolia Double Density controller.

Model TWOET 455	**Model TWOET 55F**	**NEW!**
2 Shugart SA-455 half height	Two Teac 55F half height	**HARD DISK**
48 tpi double sided	96 tpi double sided	**for**
All hardware	All hardware	**H-89**
Metal, shielded mounting plates	Metal, shielded mounting plates	**$995.00**
Data cable with chassis connector	Data cable with chassis connector	**Internal**
Power "Y" connector	Power"Y" connector	
Complete documentation	Complete documentation	
Price....... $355.00	**Price....... $385.00**	

SA-860 DS/DD half hgt 8"	505.00 2@ 495.00 ea.
SA-455 DS 48tpi ½ hgt 5.25"	140.00
FDD-100-8 SS/DD 8"	125.00
Teac 55F 96tpi DS/DD half height	155.
FDD-200-8 DS/DD 8"	**Sale 185.0**
Shugart 10 MB half height hard disk	725.00
PGS, HX-12 RGB color monitor	525.00
Data connectors of all types	**CALL**
Power connectors for all drives	**CALL**
Controllers	**CALL**
Hard disk Z150/160	Internal 750.00

Wondering what to do with your internal drive if you buy this system?

Here's the solution. If you purchase a dual half height system for your Heath computer from Floppy Disk Services, just include an extra $60.00 plus shipping and receive a single 5¼ case with power supply and data cable ready to receive your SIEMENS internal drive! the case with data cable is normally a $80.00 item. And the cable that comes with your TWOET system includes the external chassis disk I/O connector.

Due to production deadlines, this ad is 2 months old at time of printing. We encourage you to call for our latest pricing and catalog requests.

Dealer inquiries invited.

We accept cash, MasterCharge, Visa, personal checks (allow 10 days to clear the bank), wire transfers, money orders, purchase orders from government agencies, and approved businesses. Call for info.

Floppy Disk Services inc. has been supplying custom enclosures and disk drives to the hobby and professional market for 6 years now! We have specifically supported the Heath/Zenith community with a high degree of success. We care about you and your system integrity. That's why we have been a success in a time when many others are going out of business. We are proud to be a part of the Heath/Zenith community and will continue our support with new products and ideas.

H-88/89/90 and Z-150/160 are registered trademarks of Heath Corp.

GREAT PRICES!

SA-455

Shugart SA-455 half height 5¼ is the standard for excellence in the disk drive community. Backed by a one year warranty and manufactured by the leader of drive technology, Shugart Corp. Ready to ship at $140.00 ea. Compatible with ALL of the Heath/Zenith computers.

Maynard and DTC

We can supply PC compatible upgrade boards for your Z-150, Z160 system. Hard disk controllers, memory upgrade, and space saving Maynard modular boards. As an example, you can install a Maynard Modular 5 & 8 controller that will run 5¼, 8 inch and optional boards such as a clock/calendar module, game port, serial port or parallel port. Call for details on the slick way to upgrade and save a slot at the same time...

Irwin-110

Data loss! Its a terrible thought. Minimize your chance of lost programs and data files with the new 110 tape drive from Floppy Disk Services inc. It works off the standard 5¼ controller which means you save money over standard tape backups that you must buy the controller for. The data cartridge holds 10mb of data and can retrieve in streaming or individual file form. Software included. Available in many configurations. For the IBM-PC, Z-150 or Z-160 only. (CP/M software is being worked on). Tape unit $395 ea., tape cartridge $29.95 each...

Z-150/160 Upgrade

We can upgrade your Z-150/160 with half height floppies, hard disks, 8 inch drive systems, clock modules, memory and much more! Glance over our ad and call toll free outside NJ for our latest catalog. Or call and ask our sales/technical experts for their upgrade suggestions...

SA-712 & MicroScience

Half height 10 megabyte (20mb available soon) hard disk drives. The MicroScience has a plated media and is available with a full size bezel option. The Shugart has an on board low current option and is backed by a one year warranty. It also has a full size bezel option. both drives are available for the H/Z-89, H/Z-150, H/Z-160. $550.00 each.

Connectors

Floppy Disk Services inc. can supply custom data and power cables for your needs. We use only the best! UL rated or listed components, and all power connectors are crimped by our AMP electric certified crimper. No hand crimping to vary the degree of accuracy. Only the best! Floppy Disk Services inc. can make 1 or 1000 connectors to your specs, fast! Call and ask for details...

Special Sale!

Because of our enormous buying power in the industry we have made and excellent purchase of disk drives for your Heath/Zenith computer. Two Teac 55F 80 track double sided drives, in case with power supply and shielded data cable, your price $395.00. Single drive system, $245.00.

Enclosures!

We manufacture and stock enclosures for every disk drive on this page and for almost any other drive you are likely to find. Available assembled and tested with state-of-the-art power supplies, these enclosures house a wide combination of half- and full-height drives to complement most available computers.

Toll Free Order Line:
(800) 223-0306

Tech Help or Info:
(609) 799-4440

FLOPPY
DISK
SERVICES™
INC.

39 Everett Drive Bldg. D, Lawrenceville, New Jersey 08648

The Z-100 PC Technical Column (#2)

Debugging With MFM-150

Mark J. Foster
Senior Systems Engineer
Systems Software Engineering
Zenith Data Systems Corporation

Welcome back! This month, we are going to discuss some more details of the ROM based debugger, including an introduction to using it to debug programs, as well as some details of how the debugger works.

Debugging With MFM-150

As we mentioned last month, one of the nicest features of the MFM-150 monitor/debugger is its usefulness when debugging programs. This month, we'll take a quick look at the capabilities of the MFM-150 monitor, along with a discussion of some of its special features. This section is primarily intended for those of you who are writing assembly language programs, or for those who are debugging compiled programs (MFM-150 is a machine language debugger - it is not usually too helpful when writing programs in an interpreted language, such as BASIC).

To begin with, you may remember that MFM-150 was patterned after Microsoft's DEBUG utility. It operates with essentially the same commands and syntax, therefore, you don't have to learn a whole new command language to use it. With this in mind, the discussion which follows will focus on the enhancements and unique features which are provided by MFM-150.

The Help command in the monitor ("?", followed by the <Return> key) provides a summary of the commands which are available to you. The B (Boot), C (Color Bar), TEST, and V (Video) commands are unique to MFM-150, the rest are essentially the same as those provided by DEBUG. Some enhancements are present, however. If you aren't familiar with DEBUG, it will probably help to look over the DEBUG section in the MSDOS 2 manual, as well as the MFM-150 monitor description in the back of the Z-100 PC Operations Guide.

Most of the commands used in the ROM require one or more parameters to be entered. For example, to "Examine" memory you've got to enter the address of the byte to be examined (exactly the same as in DEBUG). The ROM debugger in the Z-100 PCs, though, allow you to always use register names anywhere a number could be entered. For instance, you can type `D ES:DI` to display the memory pointed to by register DI within the ES segment! This is quite handy when you are debugging programs, since you can use the actual contents of the CPU registers directly, instead of being forced to look up the values yourself.

Since MFM-150 allows you to enter register names anywhere a number is called for, you can actually do strange things like the following: Enter the monitor using Ctrl-Alt-Insert, and then type "R AX" <Return> to examine register AX. The debugger will show you the current value and prompt you for a new value. Enter a 4, followed by the <Return> key. Next, you can say "V M AX" which will tell the debugger to set the Video Mode to the contents of register AX, which in this case is 4. As a result, you'll now be in Video Mode 4 - Medium Resolution graphics mode. You can then type "VM3" to get back out. Similarly, you can try out commands like:

```
M DS:SI L CX ES:DI
```

This mouthful will move the contents of memory pointed to by DS:SI, CX bytes long, to the area pointed to by ES:DI. For you assembly language afficiandos, this is essentially the action taken by the "REP MOVSB" instruction (though the ROM is more sophisticated in that it checks for overlapping moves, etc).

To actually use the monitor to debug a program, boot up your system and obtain a directory by typing "DIR", followed by the <Return> key. As soon as the system starts to display information, quickly type Ctrl-Alt-Return. As soon as you type these keys, the monitor will show you the contents of the CPU registers, followed by the now-familiar "->" MFM-150 prompt. The register's values you see are a copy of the actual contents of the registers within the 8088 microprocessor the instant you pressed the <Return> key. Additionally, the ROM shows you the "current" instruction, which is the next instruction that will be executed, if you let the program continue.

Next, enter the "U" command (without a parameter). This will show you the next 32 bytes of code which will be executed if you don't change the instruction pointer. If you want, you can trace through the program (single-step) by entering "T5" or the like (the number you enter tells the debugger how many instructions to single-step. You can pause the display as usual by pressing Ctrl-S). Additionally, you can use breakpoints to tell the program to run at full-speed until it executes a specific instruction, AS LONG AS THE INSTRUCTION IS IN RAM. This last point must be emphasized, because all software debuggers must implement breakpoints by modifying your program temporarily. If you try to place a breakpoint in ROM, this won't work, and it will seem as though your program never executed the breakpoint, when in fact, it may well have.

In your Z-150 or Z-160, the ROM is located at F000:8000 through F000:FFFF. So, whenever you try to place a breakpoint in segment F000, it simply will not work. The best bet when you are debugging programs, which use the ROM, is to either place a breakpoint immediately following the ROM call, or to single-step through the ROM (which does work).

At any rate, you can now "debug" the DIR program. Once you are finished, you can use the "G" command to tell the CPU to execute the program being debugged at full speed. In this case, you should see the remainder of the directory on the console (assuming that you didn't modify any registers that DIR was using). This simple sequence of operations can be used with any program, including those you write: Use Ctrl-Alt-Return to stop the program and enter the debugger, perform the debugging, then use "G" to let the program continue executing.

One disadvantage of using Ctrl-Alt-Return to debug your programs is that you may not be able to type fast enough to stop the processor before it executes the code you are trying to debug! The solution to this problem is to include an "INT 3" instruction in your program. This instruction will automatically transfer control to the ROM when it is executed. There is one important note you must remember when you include the INT 3, you must manually transfer control around the instruction, or else MFM-150 will try to debug itself, and will lock up. This is really rather simple, if the instruction appears at, for example, 0AEA:97C3, you would then change the instruction pointer to 97C4 before execution (the INT 3 instruction is one byte long).

One last point is worth bringing up. The register names used by MFM-150 correspond as closely as possible to those used by DEBUG, with one notable exception. The processor's flags register is known to DEBUG as "F," and to MFM-150 as "FL". The reason for this is that you can always use register names and hex numbers interchangeably in the ROM. Therefore, MFM-150 couldn't tell the difference between a hexadecimal "F", and the flags register "F", so the name was changed to "FL".

Debuggers and Their Registers!

One question I often receive is "How can the debugger allow you to change the CPU registers, when the debugger obviously must use the actual CPU registers in order to run?." Indeed, the debugger is just another program, which does modify the CPU registers as it runs. The secret to a debugger is the use of a "pseudo-machine". This really means that the debugger cheats. When you are examining and changing the CPU registers, you aren't really modifying the real registers. Instead, you are changing memory images of the registers, which we can call "register images".

The trick to using these register images is fairly straightforward. When you first enter the debugger (either via Ctrl-Alt-Return, or when your program executes an INT 3 instruction), the debugger copies the actual CPU registers to a set of memory variables. Then, it displays the contents of these variables on the console (that's the register dump you see). Finally, it disassembles the current instruction, and prompts for a command.

Now, if you modify a register using the R command, you will actually modify the variable which corresponds to the register that you modified. For example, if you change the instruction pointer via "RIP", what will change is the variable MFM-150 knows as "REGISTERS.RIP". When you are finally finished looking at memory, etc, and you want to execute your program, the sequence is reversed. MFM-150, upon seeing a Trace or Go command, will load the register images into the actual CPU registers and will then jump into your code. The final result is that you can use the debugger just as though you were modifying the actual registers, and the ROM takes care of the details!

Next Month

The discussion this month on the debugger turned out to be long enough that I didn't have room to cover programming video modes, so we'll wait on that one until next month! Also, we'll take a look at some other useful programming information. Until then, enjoy!

Using The MS-DOS Memory Disk

Pat Swayne
HUG Software Engineer

If you are one of the lucky ones to have the latest release of MS-DOS version 2 for the Z-100-PC (Z-150) series computer, then you have a program on one of your MS-DOS distribution disks called MDISK.DVD, and instructions in your manual on how to use it. However, with previous releases and/or the version for Z-100 computers, you may have wound up with an MDISK program, but no instructions on how to use it. There is also an MDISK program on the Programmer's Utility Pack, but (at least on the release I got), the MDISK program is a new version (in source code form), with inadequate instructions in the manual on how to use it. In this article, I will attempt to clear up the MDISK mess, and tell you how to get the best use from the MDISK program you have.

The MS-DOS 2.13 MDISK

If you have MS-DOS version 2.13 for the Z-100, you will find an MDISK.DVD program on your second distribution disk, but nothing on it in the manual. This is the early version of MDISK, and it is preset to reserve 64k of your memory for use as a memory disk. (The actual space for files is less than 64k because some is used for the directory.) You load it by including the following line in your CONFIG.SYS file:

```
DEVICE = MDISK.DVD
```

If you do not have a CONFIG.SYS file on your system disk, you can use any editor (such as EDLIN) to make one. All it needs to contain is the above line (unless you want to set some other configurable parameters). When you boot up on your disk containing MDISK and the proper CONFIG.SYS, a memory disk will be installed as drive I:. This MDISK program also works on a Z-150 type computer, but the drive number becomes the next higher unused letter. For example, if you have drives A: and B:, the memory disk will be drive C:.

If you want a memory disk using an amount of memory other than 64k, you can patch the MS-DOS 2.13 MDISK program using the following chart:

Mdisk Configuration Info

Mdisk	Addresses to patch			
Size	11A	11C	11F	1AA
16K	10,0	20,0	01,0	00,04
20K	10,0	28,0	01,0	00,05
32K	10,0	40,0	01,0	00,08
64K	20,0	80,0	01,0	00,10
128K	40,0	00,01	02,0	00,20
256K	80,0	00,02	03,0	00,40
512K	00,01	00,04	04,0	00,80
	Direct. Entries	Total Sectors	FAT Sectors	Res. Space

The patches can be made using the DEBUG program. For example, if you wanted a 128k memory disk, the session with DEBUG would look like this:

```
-NMDISK.DVD
-L
E11A
xxxx:011A   10.20   00.      80.00   00.01
-E11F
xxxx:011F   0C.02   00.
-E1AA
xxxx:01AA   00.     10.20
-W
```

Notice that the old value at location 11F in the above example is 0C. This value is larger than it has to be, so even if you are going to leave the size of the memory disk at 64k, you should patch location 11F. You will have a bit more file space in your memory disk if you do.

As indicated on the table, the patches determine the number of directory entries, the total number of disk sectors, the number of sectors for the File Allocation Table (FAT), and the amount of memory to reserve for the memory disk. You can, therefore, look at the table and see how many directory entries you get with a disk of a given memory size. For a 128k disk, the table says 40,0, which is the hexadecimal number 0040, which means you get 64 entries. You can, if you want, patch in more or less directories for a given memory size

You can also make a memory disk with a size other than what is shown on the table. For example, if you want a 192k disk, add the total sectors and the reserved space figures for 64k and 128k disks from the table. Do not add the directory figures, but just choose them based on how many directory entries you want.

The Programmer's Utility Pack MDISK

The MDISK program supplied with the Programmer's Utility Pack is in assembly source form (MDISK.ASM), so you must first assemble it before you can use it. To assemble it, copy it and the following files from the Z-100 BIOS disk (even if you don't have a Z-100) to a disk with sufficient space: DEFASCII.ASM, DEFDEV.ASM, DEFMS.ASM, MACLIB.ASM, PARMS.ASM. Also, copy MASM, LINK and EXE2BIN from your distribution disks and the Utility Pack disks to the new disk. Then, you can log on to the new disk and assemble MDISK using the following example:

```
MASM MISK;
LINK MDISK;
DEL MDISK.OBJ
EXE2BIN MDISK.EXE MDISK.DVD
DEL MDISK.EXE
```

After the process is complete, you will have a file, MDISK.DVD, that you can copy to your system disk. As with the MSDOS 2.13 MDISK, you install this one with a DEVICE = MDISK line in your CONFIG.SYS file. However, this version of MDISK allows you to specify the size of the disk in the CONFIG.SYS file as follows:

```
DEVICE=MDISK.DVD SIZE=128
```

The size can be any value from 32 to 640. If you do not specify a size, a 32k memory disk is created. You can also specify a starting address for your memory disk in the CONFIG.SYS line, but that should be done only if you have some non-contiguous memory in your computer that you want to use as a memory disk. If you had 128k of memory starting at 8000:0, you could use it for a memory disk with this CONFIG.SYS command:

```
DEVICE=MDISK.DVD SIZE=128 START=8000
```

Another feature of this version of MDISK is that it tells you its drive letter when you boot your disk, with a message like this:

```
MDISK installed as drive d:
```

where d: is the drive letter.

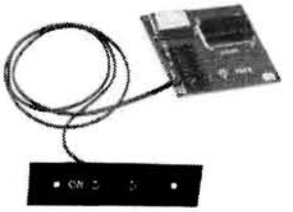
To All MDISK Users

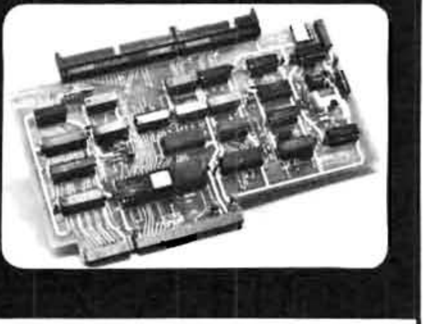
You should be aware that when you install a memory disk, it takes up some of the memory normally available to your programs. You should allow yourself at least 128k after the memory disk is loaded. That means that if you have 192k, you should limit your memory disk size to 64k. Users with only 128k total memory should probably do without, although you might be able to run your programs with a 32k or smaller memory disk installed.

A good use for a memory disk is to place WordStar and its overlays on one. Then log on to that disk and run WordStar from it. You will find that the time WordStar takes to access its overlays is negligible. If you have any programs that utilize COMMAND.COM to execute commands, you can make them work faster by including these lines in an AUTOEXEC.BAT file:

```
COPY COMMAND.COM d: >NUL
SET COMSPEC=d:\COMMAND.COM
```

where d: is the drive designation for your memory disk.





Bulletproof Garbage Filters

Some Thoughts On INPUT Routines

(For The Special Attention Of MBASIC Programmers)

Kurt A. Schultz
115-1 Roxanne Court
Walnut Creek, CA 94596

My Wife always snickers when I ask her to play "Guinea Pig" for my latest creation. I consider "Play-testing" to be an exceptionally crucial and serious part of programming games, but she thinks that I'm just goofing off. She doesn't seem to understand the processes involved in creating, refining, debugging and polishing a program, nevertheless, I refuse to remove her from the "Idiot-Proofing" tests. She can find the largest holes in my tightest programs. If I've taken my best shot at some programming problem, anything from the adequate to the elegant, she WILL find at least one thing that I've overlooked.

She isn't quite "Computer-Illiterate." It is possible that she may know more about how to run an A**** than I do. (It's true...zero plus one still isn't much!) This is why I've gotten phone calls at work asking "What's a 'SYNTAX'?," as if she wanted to add "Will you please just SHOW me one? Or tell me what it LOOKS like, so I'll know. You know, just point one out." and so on.

Having a wife who is computer-wise, "semi-literate," can really be a blessing in disguise, though. If you want your programs to succeed in the real world, you really do need to make absolutely sure that your program handles all of the pertinent possibilities. If you want a tough "beta" test of your program, have the tester be illiterate and don't give them any instructions. If you can't find anyone unable to read, at the very least, have anyone else be your tester, don't rely only on the problems that you have already thought about and solved because you've probably overlooked something.

It has often been the case that I've turned over to my tester a program that has quit "limping along"' and is "running." After she gets done with it, I find out about things I've omitted, overlooked or are unsatisfactory for other reasons. After I've fixed those, the program goes from the "running" category to the "flying" category and becomes a candidate for expansion, refinement and polish (and, of course, further testing...).

During the process of fixing program bugs, I've learned some valuable lessons, especially about getting input from "Non-technical" types. It seems to me that the one thing that you don't want is to have your program come to a screeching halt by having some blankety-blank-blank "ERROR" message crash the execution. The program MUST be able to continue execution until completion.

Assuming that your program logic is correct, can your program recover if someone tries to answer with a number when the program is expecting a letter as input? Does it accept lower case letters? How much memory are you using to execute those different answer-decoding routines? Can you do the same thing in less space if you use a different routine?

Let's see if you can avoid some of the frustrations that I have had to overcome.

****

INPUT routines have three major parts: the prompt (which will not be treated here), the section which detects erroneous input (ie: the "garbage filter," which is the main focus of this article), and the part that takes action in response to what was input (the decoding segment, the secondary focus of this article).

It is possible that the single most common letter tested for is probably "Y" (ASCII #89). I recall seeing a certain game program that could be played simply by answering YES/NO type questions. The decision-making line of code that followed the "INPUT" statement could easily look like this:

```
40 IF ANSWER$="YES" THEN 200
```

This is the "adequate" category for decoding a User's input, since it has significant limitations. What happens, for example, if our overly-eager hunt & peck typist answers with "YEA" instead of "YES," an error of only one letter, with the keys for "A" and "S" right there next to each other? The intent of the answer has not changed, yet the program's flow will be vectored to an unwanted area, because computers interpret instructions literally.

For the convenience of the User, then, the programmer needs to add some depth to the acceptability of responses. One might do it by adding an extra statement line for each of the extra responses allowed and tested for. Then the code section might look like this:

```
40 IF ANSWER$="YES" THEN 200
50 IF ANSWER$="YEA" THEN 200
60 IF ANSWER$="YUP" THEN 200
70 IF ANSWER$="YEAH" THEN 200
```

This isn't getting much done for the effort...it will work, but yuck, icky-glop and other such comments. I don't like this kind of a "ladder" routine, mostly because it wastes space in RAM, and I, therefore, try to avoid using it. About the only place I want to use this method is during the development of a program, when I am not sure if some of the "acceptable" answers will be changing as the program develops. Once the algorithm has been finalized, these kinds of routines are replaced with equivalent code which is more elegant, during the development stage called "polishing."

Anyway, in our example, I see a pattern developing. All of the "yes"-type responses seem to start with a "y". No, excuse me, it's a "Y", as in upper case. Also, why have all of those lines of code when one line could test for all of those inputs?

We could combine the different acceptable choices with Boolean Operators. This would transform the previous section of code into something like this:

```
40 IF ANSWER$="YES" OR ANSWER$="YEA" OR ANSWER$="YUP"
   OR ANSWER$="YEAH" THEN 200
```

This line of code does use less RAM, but it's not easy to read. It is also quite a mouthful to say in one breath. In fact, it can be done a little more elegantly if we only test the first letter of all responses. The addition of the (previously omitted) statement line that fetches the answer, the addition of another statement line and the refinement of the testing statement line gives us something like this code segment:

```
20 INPUT "DO YOU WANT INSTRUCTIONS ";ANSWER$
30 ANSWER$=LEFT$(ANSWER$,1)
40 IF ANSWER$="Y" THEN 200
```

After the input is fetched from the keyboard, everything except the left-most letter is thrown away. What remains is then tested against the character "Y". This segment will accept as affirmative any response that starts with a "Y" character. Are you aware of any negative-response expressions that start with a "Y"? (YO MAMA! does...)

But, what if your User is from the Use-'1'-for-'YES'-and-'2'-for-'NO' School of Adequate INPUTS? There actually were some early dialects of BASIC which did not allow for string variables (anyone remember why they issued Extended Benton Harbor BASIC?), therefore, all responses HAD to be numeric. The User's habit may be to answer "1", which for this section of code would not be proper and would result in the unintended. This problem is also true for expressions like "Affirmative," "Well," "Maybe" and "Of course!."

This segment of code is still limping. What is needed is a test to restrict the responses to those that are important, one that would "loop back" for new input if it detected any unimportant responses. In essence, a garbage filter. This requires that we define just exactly what responses are acceptable:

1) Any response that starts with a "Y"
2) Any response that starts with a "y"
3) Any response that starts with an "N"
4) Any response that starts with an "n"

This will, of course, cause the second test (the one on line 40) to lose more than a few (in fact nearly ALL) of its potential responses, but it has the greater benefit of preventing the program from getting confused as easily (most of the potential responses are insignificant anyway). It will also allow for lower case answers. The statement line added is:

```
35 IF NOT(ANSWER$="Y" OR ANSWER$="y" OR ANSWER$="N" OR
   ANSWER$="n") THEN 20
```

This test will pass all responses that start with "y" or "n," both upper and lower cases, and fail everything else. Our segment of code now looks like this:

```
20 INPUT "DO YOU WANT INSTRUCTIONS ";ANSWER$
30 ANSWER$=LEFT$(ANSWER$,1)
35 IF NOT(ANSWER$="Y" OR ANSWER$="y" OR ANSWER$="N" OR
   ANSWER$="n") THEN 20
40 IF ANSWER$="Y" THEN 200
```

Once the program has a response that isn't garbage, the response is tested to see if it is an affirmative one and execution control is sent to the proper place, except that it still won't pass a lower case "y" until we add "OR ANSWER$="y" " to line 40, just before the term "THEN". After this change, it looks like this:

```
20 INPUT "DO YOU WANT INSTRUCTIONS ";ANSWER$
30 ANSWER$=LEFT$(ANSWER$,1)
35 IF NOT(ANSWER$="Y" OR ANSWER$="y" OR ANSWER$="N" OR
   ANSWER$="n") THEN 20
40 IF ANSWER$="Y" OR ANSWER$="y" THEN 200
```

Do you think that this is about the best we can do for a yes/no response filter? No, not by a long shot. There are some other things that could happen, as well. (See my article "A Problem and Its Solution" for one example.) Since I'm not satisfied with the way that the lines are numbered, I'm going to combine lines 20 and 30, thereby allowing line 35 to be called line 30. Finally, the code segment looks like this:

```
20 INPUT "DO YOU WANT INSTRUCTIONS ";ANSWER$:
   ANSWER$=LEFT$(ANSWER$,1)
30 IF NOT(ANSWER$="Y" OR ANSWER$="y" OR ANSWER$="N" OR
   ANSWER$="n") THEN 20
40 IF ANSWER$="Y" OR ANSWER$="y" THEN 200
```

Later in this article we will see a method which is much better.

****

Looking at some alternative input requirements, it looks like there are many filtering and decoding problems that can't be solved with simple yes/no gates. One of the programs I'm doing requires the User to select a choice from a menu. The choices are: C, D, E, F, H, M and T (for Cleric, Dwarf, Fighter, Halfling, Magic-user and Thief, respectively). The sections of code that I used look something like this:

```
      .
      .
      .

110 CLASSIC$="CDEFHMT"

      .
      .
      .

930 INPUT "CLASS ";CLASSY$:CLASSY$=LEFT$(CLASSY$,1)
940 FOR I=1 TO 7:IF NOT(CLASSY$=MID$(CLASIC$,I,1))
    THEN I1=1:GOTO 960
950 I1=0:I=7
960 NEXT I:IF I1=1 THEN PRINT BLINK$;:GOTO 900

      .
      .
      .
```

First, in line 110, I defined a string variable to equal the first letters of all the valid options. This string variable is the "key" that I will later test against. Next, I displayed the menu (lines 900 to 920, not shown) and fetched the input (line 930), assigning the input to the variable CLASSY$. Since I am only interested in the first letter of the response, I throw away everything except the leftmost character (also line 930). Line 940 contains a FOR/NEXT loop, which tests the content of the now-modified response (CLASSY$) against each of the characters in the string CLASSIC$. If there is no match, set a flag (I1) and try again with NEXT I, which is to say, the next character in CLASSIC$. If all of the tests fail to match, the flag will remain set after the last test. If there is a match, reset the flag, set the index variable to its maximum and then exit the loop via the "NEXT I" statement.

The method in which this is set up is quite critical. The Interpreter will only allow one "NEXT" per "FOR," therefore, the existing "NEXT" must serve as the common exit point for both possible outcomes of the "IF" statement. After the loop has been exited, test to see if the flag was set by the loop, and branch execution accordingly. In this case, if the flag "I1" is set (equals one), then the routine searched through all of the characters in CLASSIC$ without matching. The string variable BLINK$ contains escape sequences to erase the screen and position the cursor, as well as an error message indicating the problem. After "BLINK"ing, the program loops back to re-display the prompt and then fetch another input.

Eventually, an acceptable input will be entered, the test will match, the flag will be reset, the index variable will probably be altered (the index will remain unchanged if response matches during the last test), the FOR/NEXT loop exited and the flag will be tested. When something has matched, the flag will equal zero; therefore, execution control will fall through to the next line, successfully passing through the "garbage filter."

The application that this example was taken from only required that the input be "legal," action (determined by the value of the choice) would be taken later, by another routine. Meanwhile, the responses were to be stored in a disk file. A decoding routine wasn't needed immediately, but the garbage filter was. There is an advantage to using the arrangement just demonstrated (ie: scanning a "key"), in preference to a "ladder," especially if the "legality" test has to handle several times that many possible inputs. If you use this approach, just make sure that all of the possibles are represented in the "key" string, that the FOR/NEXT loop's range (or number of times through the loop) is equal to the number of characters in the "key" and that the value used in the statement which modifies the index's value is equal to the limit of the FOR/NEXT loop's index. In other words, if there are 7 options, then there are 7 letters in the string variable CLASSIC$, there are 7 passes (or less) made by the FOR/NEXT loop and to force a quick exit from the loop, set its index equal to 7, then exit via "NEXT I." Equivalently, if there are 12 options, then there are 12 letters, 12 passes, and setting the index to 12 prior to executing the "NEXT I" will cause an early exit from the loop.

There is a more elegant way of accomplishing the same thing. In fact, I think that it is so sweet it should be saved for dessert. Before we get to that, though, I want to "serve up" some different aspics, excuse me, aspects of these problems.

****

Another program from the same package presents us with a different problem. It requires that the inputs be a single letter, A-Z or a-z, inclusive. This happens just after an alphabetically-encoded menu is displayed. However, on the menu, the last few letters have no meanings assigned to them (there probably will be meanings there as the program develops). So, once we fetch the input and place it into a string variable, we again throw away everything except the leftmost character of the input. The code segment looks like this:

```
450 INPUT "What is your choice ";CHOICE$:
    CHOICE$=LEFT$(CHOICE$,1)
```

Now, what we have to do is to convert, if needed, lower case responses to upper case responses. If the response is lower case, it will have an ASCII value greater than or equal to 97 (which is to say greater than 96), but less than or equal to 122 (ie: less than 123). The way to convert these inputs is to subtract 32 from the ASCII value of any character with an ASCII value greater than 96. Encoded, we get the next line:

```
460 IF CHOICE$>"'" THEN CHOICE$=CHR$(ASC(CHOICE$)-32)
```

Notice that "garbage" responses with an ASCII value greater than 122 are also reduced by 32. This is not a problem, since the next step will deal with them in the same manner that it deals with ASCII values 91 through 96 (123-32=91, etc.).

If the conversion is or is not needed, the next thing to test for is whether or not we have received valid inputs. Everything above ASCII 97 will have had 32 subtracted from it, therefore, any input that is less than or equal to ASCII 64 is illegal, and so is anything greater than or equal to ASCII 91. We can encode the logic thusly:

```
470 IF CHOICE$<"A" OR CHOICE$>"Z" THEN PRINT BLINK$;
    :GOTO 320
```

or this way:

```
470 IF ASC(CHOICE$)<65 OR ASC(CHOICE$)>90 THEN PRINT BLINK$;
    :GOTO 320
```

when we understand that "BLINK$" is similar to that used in the previous example and the "GOTO 320" statement sends control back to the beginning of the prompt display. Of the two, the first line seems to require less RAM and is therefore the preferred method.

Grouping those lines together, it looks like this:

```
450 INPUT "What is your choice ";
    CHOICE$:CHOICE$=LEFT$(CHOICE$,1)
460 IF CHOICE$>"'" THEN CHOICE$=CHR$(ASC(CHOICE$)-32)
470 IF CHOICE$<"A" OR CHOICE$>"Z" THEN PRINT BLINK$;
    :GOTO 320
```

A quick recap shows that when line 470 has been completed, the following things have occurred:

1) The prompt has been displayed (lines 320-440 and part of line 450).
2) Nearly any kind of input has been accepted and everything, except the leftmost character of the response, has been discarded (line 450).
3) Lower case responses have been converted to upper case (line 460).
4) Unacceptable responses have been taken care of (line 470).

Now the program needs to decode those responses. We know that the response has an ASCII value somewhere in the range of 65 to 90, inclusive. Since the application requires that most of the responses be handled in a different manner (although some will be handled in an identical manner), each response must cause a branch to one of several different handling routines. It appears that the easiest way to do this is to use the "ON/GOSUB" instruction. For the application from which this example was taken, each subroutine is no more than one program line long and each subroutine sends control, when done, to an ending point that is common for this entire set of handling subroutines, a "RETURN" statement. The ASCII value of the response needs to be offset in order to map an "A" response into the first handling subroutine, a "B" response into the second subroutine, and so on. There are no responses that have ASCII values less than 65 or greater than 90, since we've already filtered that garbage, so the number of line #s in the ON/GOSUB command's list of line #s will be 26, and there is no need to handle the situation where the value of the expression equals zero or is greater than 26. Once we see that the handling subroutines start at line 2000, we can see that the decoding statement line must look something like this:

```
480 ON ASC(CHOICE$)-64 GOSUB 2000,2010,2020,2030,2040,
    2050,2060,2070,2080,2090,2100,2110,2120,2130,2140,
    2150,2160,2170,2180,2190,2200,2210,2220,2250,2250,
    2250
```

Notice that the last three branches are all to the same line. The handling subroutine on that line responds to the choices "X", "Y" and "Z" with the same statement: "This choice is not yet implemented...choose again." Because there is the possibility of needing to select several of the menu options, line 490 is the start of a YES/NO gate that checks to see if the menu should be redisplayed and another choice allowed.

If the application didn't require separate handling subroutines, but instead required those menu options to, let's say, only toggle some flag in a one-dimensional array, then line 480 might look something like this:

```
480 FLAG(ASC(CHOICE$)-64)=FLAG(ASC(CHOICE$)-64)*(-1)
```

The status of the flag is then dependent upon whether the corresponding array value is positive or negative.

****

Another input problem that occurs in programming is this: How do you get your program to fetch a password? A password should not appear on the screen, therefore, it needs to be entered through the keyboard without being echoed to the CRT. Fortunately, there is an instruction in the command set that can be used for just this occasion. The command is "INKEY$".

There's a couple of tricks to using this one, though. To begin with, the response that it usually returns is null (ASCII 0), which isn't really worth anything. ("Eureka, I've discovered the NULL!" "What's that?" "Oh, nothing, nothing...") This forces you to keep fetching a response from the keyboard until you get something that is useable. If that's the case, then you may as well keep executing the statement line that has the "INKEY$" instruction in it, so after you get a response, test to see if the response is null and loop back if it is.

But hold on a moment, now! How many times is the program going to fetch a response, test it for "nullness" and loop back to fetch again, while it's waiting for this slow, analog, organic operator to decide which key to depress? One to ten MEGAGOBS is my guess. Know what this means? It means that under no circumstances should you have any kind of display, such as a prompt, going to the CRT from inside of this kind of a loop. If you do, your prompts may scroll off the top of the screen faster than an icecube melts when placed on an Arizona sidewalk in mid-summer! Even if you have a command that always pre-positions the cursor before displaying the prompt, you will have an irritating flicker problem, so don't even try it! The way to avoid these problems is to display a prompt on one line, then have the "INKEY$" command be in the first statement on the next line, and when you loop back, go to the line directly after the prompt. It is too bad that I didn't read any articles like this one before I encountered that problem.

But enough of these warnings, let's get back to the matter of fetching passwords. The code segment that I usually use looks something like this:

```
         .
         .
         .

780 PRINT "Password (7 Characters MAX) ?";
    :REM CREATES A 1-LETTER TO 7-LETTER PASSWORD
790 WORD$="":FOR I%=1 TO 7:REM
800 A$=INKEY$:IF A$=CHR$(13) THEN I%=7:GOTO 830
810 IF A$<CHR$(32) OR A$>CHR$(90) THEN 800
820 WORD$=WORD$+A$
830 NEXT I%:WORD$=WORD$+STRING$(7-LEN(WORD$),32):PRINT

         .
         .
         .
```

Line 780 gives the input prompt. Line 790 erases whatever "WORD$" may have contained and also starts the loop. Line 800 fetches one character from the keyboard and tests to see if it is a carriage return (ASCII #13). If there is nothing waiting to be fetched from the keyboard (almost always the case), the "INKEY$" function will return a null (ASCII #0) and set "A$" to that. If the character returned is a carriage return, then the user must have completed the entry of his(her) password, so the loop's index gets set to its max and then control heads to the "NEXT" statement, forcing an early exit. The next line, 810, filters all of the remaining possible responses for garbage, and if any is detected, sends control back to fetch another character. It is line 820 where the password is actually built. "WORD$" is constructed here, one character per pass of the FOR/NEXT loop. The last line of this sequence, 830, closes the loop and fills in (or "pads"), if needed, any unused character positions in the space allotted to "WORD$" (ie: for passwords smaller than 7 letters). Just before the end of this routine, another PRINT statement is executed, so that the next output to the CRT stands by itself.

Keep in mind that this section of code only fetches the password, it does not store it, nor does it test a suspected password against the true password. Storing and matching functions are left as exercise problems for you to work on, since we are supposed to be dealing with input routines here.

****

One area that I haven't yet touched on is that of numeric inputs. Quite simply, I don't use them. Well, let me qualify that statement: I don't assign inputs directly to numeric variables; occasionally, I do use numeric inputs. When I do, however, I always assign the input to a string variable, at first, and convert to numeric format (if I need to) later. Why do I go to all of this trouble? Most of these garbage filtering techniques work best with string variables.

Let's face it, if the interpreter signals an error because a letter or a punctuation symbol was accidentally entered when the program was expecting only a number, execution will pause at a place before the garbage filter is reached! A Programmer can't properly guard against something that prevents the execution of the guarding and filtering instructions.

This situation is very much like what happens for certain other errors that might occur during the run of a program. If the interpreter halts a program because of an error, such as DIVISION BY ZERO, the execution of that program will completely halt before the garbage filter can be reached. But wait! This, too, is a type of garbage filter and it, also, can be dealt with.

An "ON ERROR GOTO" statement needs to go somewhere in front of the input routine and needs to refer to a line where "ERR" is tested against the number 11. If the match is positive, give an appropriate warning beep and display, then "RESUME" to fetch the input again. If ERR isn't equal to 11, other conditions can be tested for or the program can "RESUME." It could look something like this:

```
100 ON ERROR GOTO 5000
         .
         .
         .
```

```
5000 IF ERR=11 THEN PRINT CHR$(7)+
     "Your inputs have caused a 'Division by Zero' error.
     Please use better data and try again."
5010 RESUME XXXX
       .
       .
       .
```

(where "XXXX" is the line number that marks the start of the input routine.) An alternative for line 5010 might be "RESUME NEXT," which allows additional versatility.

This technique can also be used for some of the other error conditions that a user might stumble into, such as #13 or #17. Error code #13 (TYPE MISMATCH) can be handled in the same manner as above, but #17 (CAN'T CONTINUE) should close any files that might be open and chain to the program that was just running (itself), thus loading an uncorrupted version of the program.

****

Now that I've explained some of the different aspects of the problem, I want to show an elegant solution or two. Many of the functions referred to earlier can be condensed into an even better way. The function "INSTR" gives the programmer a valuable and powerful string searching capability, the ability to search a key string to see if any part matches against the fetched INPUT. The key string, then, for a simple YES/NO gate is "YN," and to search the key, the command form is:

```
BRANCH%=INSTR(YNKEY$,GUESS$)
```

Going back to the original problem as an example, let's take a look at the problem with this new tool in mind. First of all, the key needs to be defined:

```
10 YNKEY$="YN"
```

The next things to do are to display the prompt, fetch the input and trim the response down to one character. Therefore:

```
20 INPUT "Do you want instructions ";ANSWER$:
   ANSWER$=LEFT$(ANSWER$,1)
```

Next, search for a match with any character in the key:

```
30 BRANCH%=INSTR(YNKEY$,ANSWER$)
```

And then take appropriate action:

```
40 ON BRANCH% GOTO 60,70
```

If there was no match, the value of "BRANCH%" will be zero, causing the execution control to fall through to line 50, therefore:

```
50 PRINT "Please answer with a 'Y' or an 'n'.":GOTO 20
```

If ANSWER$ equals "Y," then BRANCH% equals 1, so the ON/GOTO would route program control through the first option/branch. This means that the statements on line 60 must be the "Y" response handling routine. So:

```
60 GOSUB 10000
```

The subroutine starting at 10000 will do the actual explanations, of course, and after returning, the program will continue on with the line of code that is the destination for those times that ANSWER$="N".

Listed together in one place, we have:

```
10 YNKEY$="YN"
20 INPUT "Do you want instructions ";ANSWER$:
   ANSWER$=LEFT$(ANSWER$,1)
30 BRANCH%=INSTR(YNKEY$,ANSWER$)
40 ON BRANCH% GOTO 60,70
50 PRINT "Please answer with a 'Y' or an 'N'.":GOTO 20
60 GOSUB 10000
       .
       .
       .
```

If lower case responses are to be considered, the situation is hardly much more complex, but there is a catch. When the Key is defined, a space needs to be added into the front of the string as a "buffer." Actually, nearly any character that is not a valid choice will work as the "buffer," but a blank is recommended for the sake of readability. Line 10 now looks like this:

```
10 YNKEY$=" YyNn"
```

All the rest of the code and logic remains unchanged, except for line number 30, which look like this:

```
30 BRANCH%=INSTR(YNKEY$,ANSWER$)\2
```

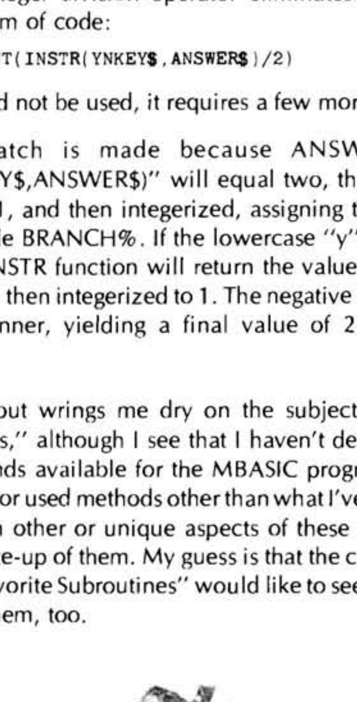
The main component of this section of code is line 30, which searches each character in YNKEY$ to see if ANSWER$ matches it. If a match is made, the function "INSTR" returns an integer value equal to the matched character's position in the string, and this value is divided by two, with the Integer Division operator (indicated by the symbol "\"), before being assigned to the variable BRANCH%. Use of the integer division operator eliminates the need for the equivalent form of code:

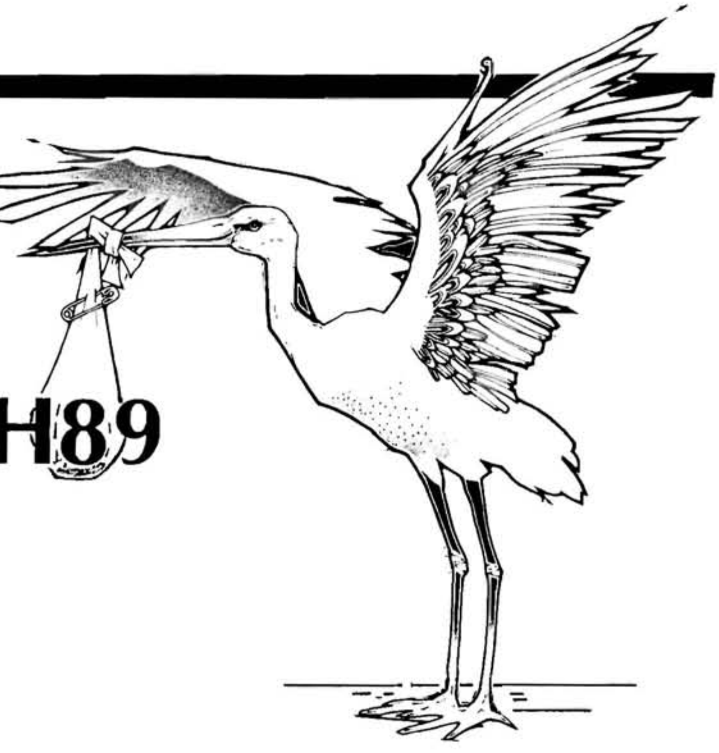
```
30 BRANCH%=INT(INSTR(YNKEY$,ANSWER$)/2)
```

This form need not be used, it requires a few more bytes of RAM.

So, if a match is made because ANSWER$='Y', then "INSTR(YNKEY$,ANSWER$)" will equal two, then be divided by two to equal 1, and then integerized, assigning the value 1 to the integer variable BRANCH%. If the lowercase "y" response is used instead, the INSTR function will return the value 3, which will be divided to 1.5, then integerized to 1. The negative responses work in the same manner, yielding a final value of 2 for the variable BRANCH%.

Well, that about wrings me dry on the subject of "Bullet-Proof Garbage Filters," although I see that I haven't dealt with all of the input commands available for the MBASIC programmer. If any of you have seen or used methods other than what I've shown here, or if you have seen other or unique aspects of these problems, please send me a write-up of them. My guess is that the column in REMark called "My Favorite Subroutines" would like to see them, as well, so send 'em to them, too.





A Born Again H89

Robert W. Hootman
10420 Mackinaw
El Paso, TX 79924

Say you want to take advantage of the "Public Domain" Software?

Say you want to take advantage of those "Generic" CP/M software sales?

Say ZDS centers are clearing out H/Z software on 8 inch at very low prices and you would like to take advantage?

Say you would like to be "Disk Compatible" with most CP/M 8 bit machines around?

So you say yes to "All the above" BUT . . . . .

Oh! You don't have 8 inch drives? What's-a-matter? Cost too much? Or maybe the right side is full and you can't expand more.

Perhaps you plan to get a new "Whiz-Bang Gee-Whiz" computer later and want to remain compatible with your drives.

I wanted "all the above" also. With a very small budget for my play pretties, I had about given up hope of seeing 8 inch drives spinning merrily away on my H89. The cost of another controller, plus the cost of the drives. made total cost a bit steep for my means. I could have obtained an H47 set while they were on sale, but that meant giving up either the H-17s or the H-37s. I've grown very fond of those drives and don't have the heart to part with them. Most of my H/Z software is on 5-1/4 hard-sectored distribution disks.

How does adding two DS/DD 8 inch drives and keeping your H-17 AND H-37 drives and controllers for less than $700.00 sound? Of course, you could add one or two SS/SD 8 inch drives for even less. I now have my three H-17s (hard-sectored) and two 96 tpi (soft-sectored) DS/DD drives AND two DS/DD 8 inch drives running on my dear ol' '89. AND I'm using just the H-17 and H-37 controllers. Wanna hear more? Just read on.

In REMark Issue #40, Mr. E.D. Blayer modified an H-37 controller for 8 inch operation. Unfortunately, the hardware changes made it useless for the 5-1/4 inch drives. The 5-1/4 inch drives require 1 MHz and 2 MHZ clock and VCO signals, while the 8 inch drives require 2 MHz and 4 MHz clock and VCO signals, thus a big problem is created.

But in "HUGGIE" land no problem remains big for very long. Enter Major George W. Mayes. He sighted problem-sank same. With a simple little plug-in paddle board for the H-37 controller board, he enables the changing of clocks and VCO frequencies "on the fly" with drive select. With this small modification and some simple changes to the standard H/Z Bios, he was able to read and write SS/SD 8 inch disks as drive "C" on the H-37 controller. He also developed a "FORMAT8.COM" for formatting the standard IBM 3740 format. Thus, with a simple modification, a SS/SD 8 inch drive, a little "Bios" change and a couple hours work, he was compatible with most of the CP/M world.

This alone was enough to gladden the heart of any "HUGGER." But wait . . . . . Enter Mr. David L. Ritter. He liked the George W. Mayes modification, too. But, he took it even further. If you can do one, why not more? What about double-sided? And even double-density maybe?

Just a minute you say? What's that? Oh yeah! You're right. The H89 just isn't fast enough to run 8 inch drives in double-density. Well . . . . What about that 4 MHz modification of Pat Swaynes? Remember that? Yup! That does it. Way back in REMark Issue #34, Pat Swayne had an article that provided a 4 MHz up-grade for the H89.

Mr. David Ritter tied both of these modifications together with some outstanding software. This allows you to run up to four drives on the H-37 controller. It also provides enough information on "how to" patch your bios, that you can run about any type of drive you desire. But in my case, I'm running two 8 inch drives as "C and D" and two of my 96 tpi 5 inch drives as "A and B." As yet, I cannot "BOOT" from the 8 inch, however, David has said he is working on it.

Now, how I went about it.

First, I contacted George Mayes as to the existence and availability of his modification board. Yes, it did exist and was available. I then recalled Priority One Electronic had been offering the SEIMEN eight inch drives with power supplies and cabinet on special. So, I gave them a call. Yes, they were still on sale. In fact, they had a pair of SEIMEN FDD-200-8's with cabinet, power supply and fan for just $597.00, plus shipping. These are the DS/DD drives. I ordered on the spot. With shipping (UPS), the total was $612.00.

Then I went ahead and ordered the George Mayes modification and the David Ritter software. Then sat back and waited. Time passes so slow when you wait for something like this.

At last it begins to arrive. Then I learn a valuable lesson. Don't plan on part houses. After many trying times and phone calls trying to obtain the IC's for both modifications, I ended up buying them from the Heath Co. They were the only suppliers that had all required IC's on hand. Also, they were quite a bit more costly. Now, all parts on hand, I was ready to go.

It is never a good idea to install more than one modification at a time. Always install one, check it out to be certain it works, then go on to the next. I didn't have the 4 MHz mod yet. I decided to start with the controller mod first.

Following George's documentation was a snap. The paddle board was easy to assemble and install on the H-37 controller. The only thing that required any head work was making the interface cable. You must go from a 34 pin connector on the H-37 controller to a 50 pin connector on the 8 inch drives. Again, George's documentation made this quite easy. Then a few simple changes to H/Z "BIOS.ASM," did a makebios and I was reading an eight inch disk, SS/SD, I had received with H/Z software. Oh! Stay still my fluttering heart. What joy! Pure rapture! Now, I know I'm in the ball-park, so on to the next.

Back to REMark Issue #34 for the 4 MHz modification. But mine is an H89A and Pat's modification is for an H89. They are not physically the same. Again, REMark to the rescue. In REMark Issue #45, Gary Wintergerst presents a plug-in module of Pat Swayne's modification for the H89A. Very simple to build and install, thanks to the well written articles. Fired up and played around a bit. Sure makes a big difference.

Now, the ultimate test. With WordStar up in the Non-Document mode, loaded in David's "BIOS033.ASM" to set the "EQU's" for my system. Ran "MAKEBIOS," and nothing worked. Several hours of trying many different things and still nothing. Finally I broke down and gave David a call. It's tough to find out you don't really know as much as you think you know. He proceeded to set me on the straight and narrow path of CP/M. This is the way I found that you must tell "MOVCPMXX" to look at drive X: for the size of the new BIOS.SYS. You must use MOVCPMXX * X:BIOS.SYS. After this bout with the manuals and David, I did everything all over again, but right this time. Loaded in the new BIOS, ran CONFIGUR and was blessed with a reborn H89A. It is almost beyond dreams to run DIR on these drives and come up with 1360k free space on each drive.

Running WordStar off the 8 inch drives at 4 MHz, as I am at this moment, is beyond my wildest dreams. The speed of these drives, when formatted in double-density, is unreal. Couple that speed with the CPU running at 4 MHz and the results can only be described as WOW!.

All of the above applies only to CP/M. I have HDOS, but use it very little. I won't even attempt to discuss the merits of, pro/con, each operating system. However, in the article by E.D. Blayer, Remark Issue #40, are patches for HDOS that should work or are close enough they can be made to work, since all ports and address are the same. That would be a nice project for the more handy than I at programming. Any takers?

Please, If you order or request information from these folks. Have the courtesy to enclose a self-addressed, stamped envelope. The price of postage these days will drive many not to answer mail even if they desired to do so.

Major George W. Mayes
6717 Welton Drive NE
Albuquerque, NM 87109

Mr. David L. Ritter
1244 Cascade Drive
Oak Harbor, WA 98277

I hope this will help somebody up-grade to the 8 inch drives. I know I have long wanted to do this, but was unaware that all this information was available. Hope this will shed some light.

Enjoy!

H/Z – 19/89

New from Apex Systems Technology, a conversion kit which will enable the owners of the H/Z – 19/89 computers to separate the keyboard from the main screen. This conversion kit is designed for comfort and flexibility when operating the computer. When completed you will have an attractive, versatile computer.

Vendor: APEX SYSTEMS TECHNOLOGY
P.O. Box 2231
Canoga Park, Calif. 91306
Price: $50.95 + $2.00 S & H
(Calif. residents add sales tax)

A Menu For Z-100 Winchester Files

Program Author: Stephen V. Chiavetta, III

Documentation Author: Stephen V. Chiavetta, MD
2512 Ridge Road
Raleigh, North Carolina 27612

INTRODUCTION:

This program, written in ZBASIC for the Z-100, requires MS-DOS 2.1 operating system and a Winchester drive. The program was written for a color monitor but can be used on a monochrome monitor without difficulty. The program will display a menu from which you select the application program you wish to run. If you are like many Z-100 users whose Winchester is loaded with files and from time to time have difficulty remembering the logical drive or directory where the program is located, then this program is designed for you.

The program begins on boot-up after entering the date and time. Then a menu of application programs are listed for selection. Each application program is listed as a numbered choice within a unique directory. The user can define as many choices as he wishes by increasing or decreasing the number of directories that appear in the menu. Since many Z-100 users are familiar with ZBASIC, changes can be made to the program easily to suit the individual's own files on the Winchester. The program combines the power of MS-DOS 2.1 operating system commands in an autoexecute batch file with ZBASIC.

It is first necessary to name your directories on the E: (and F:) drive(s) of the Winchester with the operating system using the (MD) Make Directory command. Then copy the application program files and data files to the directories you deem appropriate. After entering the date and time on boot-up, the autoexecutive batch file refers to a ZBASIC file called DOSMENU.BAS. This file allows you to select the application program file you wish to run.

THE Z-BASIC PROGRAM: "DOSMENU.BAS"

The menu choices should be changed to match your own application programs you have on your Winchester disc. This can be accomplished by simply changing the print statements listed in lines 70-190. You can create as many choices as you wish by adding or subtracting from the 12 choices listed on the menu. The names you list in the menu need not be the same as the application program file names.

Lines 220-340 delete the data files that may have been opened during a previous session. The number of data files must match the number of choices you have on the menu. To do this, you must add or subtract the number of "ON ERROR GOTO xxx:KILL 'x.DAT' " statements to match the number of choices you have on the menu. This part of the program is essential to eliminate these files from the disk, so the autoexecute batch file does not select the previous choice from the last time the program was run.

Lines 350-470 selects the number you chose from the menu. The input statement on line 200 assigns the variable "A" to the number you selected. As a result, the IF...THEN statements select which line of the program to refer to. Line 480 will cause a beep and recycle the menu choices if an incorrect number is selected from the menu.

Lines 490-970 open a data file that corresponds to the number selected from the menu. The variable "A" is written in the file to make the file contain data and therefore exist. The file is then closed. The number of data files opened and closed must be the same as the number of choices in the menu. The system command at the end of each data file command exits the ZBASIC portion of the program.

THE AUTOEXEC.BAT PROGRAM:

The date and time commands reside in the MS-DOS operating system program when the system is loaded. The first path command is written to allow the operating system to find the assign command in a directory called 'bin'. The second path command finds the files in a subdirectory called word. The assign command enables the operating system to access the second logical drive F: on the Winchester that contains application programs. The E: drive is automatically accessed on boot-up.

The PSC is a file name in the operating system that allows the screen to be dumped to the printer after executing a shift-F12. A screen dump is not part of the program but is written in the autoexec.bat file so that anytime this feature is called on it will function. You may add any of the operating system commands you like in the beginning or end of the autoexec.bat file and it will not interfere with the operation of the program.

DOSMENU is the file name of the ZBASIC program that is described in detail above. In the next series of commands (i.e. IF EXIST E:1.DAT GOTO A) the program looks to find a data file that exists. In this case a data file exists if the variable A has been written in the file by the DOSMENU program. Since only one data file exists (because only one choice is possible in the menu of DOSMENU) the GOTO statement selects the appropriate directory to open. Each GOTO statement is followed by a different variable that corresponds to a directory that contains the application program you wish to run.

In the next series of commands used to access a directory, each starts with :<variable> that corresponds to the variable after each GOTO statement. In the first group of commands beginning with :A, the CD\BIN command changes the directory to BIN, therefore, allowing the files in that directory to be accessed. The directory names listed in the autoexec.bat file must be identical to the names you have assigned to the directories when they were named using the make directory command. The second line, DIR/W/P, instructs the operating system to list the files across the screen a page at a time. The second group of commands starts with :B and does a similar function.

The third group of commands, beginning with :C, does a slightly different function. The CD \LOTUS command is followed by the application program called LOTUS. This serves the same function as typing in the file name LOTUS at the system prompt. Each group of commands all end with a GOTO Z statement. This causes the autoexec.bat file to jump by all of the nonexistent data files. The last statement in the autoexec.bat file is a customized feature of the prompt command. The command "prompt Current Directory is p—ng" tells the operating system to display the current directory, skip a line and display the prompt "E>" on the screen whenever the system prompt appears. This enables the user to keep track of where he is in the tree of directories.

WORDS OF CAUTION:

The autoexec.bat file, ZBASIC, and DOSMENU.BAS must be located in the root directory of the default boot-up partition of the Winchester drive for the program to run. The default boot partition for the Z-100 is set at the factory for the E drive. On boot-up the operating system looks for an autoexec.bat file to run. When it is found, the functions are carried out without keyboard entry.

DOSMENU.BAS

```
10 CLS
20 LOCATE 2,27:PRINT "MENU OF PROGRAMS"
30 LINE (0,0)-(639,224),1,B
40 COLOR 7,0
50 LOCATE 4,15:PRINT "Directory for E:"
60 'LISTING OF THE DIRECTORIES ON THE WINCHESTER
70 PRINT:PRINT:LOCATE 6,15:PRINT "1) MS-DOS 2.1 files"
80 LOCATE 7,15:PRINT "2) Utilities"
90 LOCATE 8,15:PRINT "3) Lotus"
100 LOCATE 9,15:PRINT "4) Watchword"
110 LOCATE 10,15:PRINT "5) Watchword
      - Steve's documents"
120 LOCATE 11,15:PRINT "6) Watchword
      - Janet's documents"
130 LOCATE 12,15:PRINT "7) Watchword
      - John's documents"
140 LOCATE 13,15:PRINT "8) Watchword
      - Stephen's documents"
150 LOCATE 15,15:PRINT "Directory for F:"
160 LOCATE 17,15:PRINT "9) Games"
170 LOCATE 18,15:PRINT "10) Basic"
180 LOCATE 19,15:PRINT "11) Basic - Steve"
190 LOCATE 20,15:PRINT "12) Basic - John"
200 COLOR 7,1:PRINT:LOCATE 22,19:INPUT "Please
      enter the number of your choice";A
210 COLOR 7,0
220 'DELETES THE PREVIOUSLY CHOSEN DATA FILE THE
      LAST TIME THE PROGRAM WAS RUN
230 ON ERROR GOTO 990:KILL "1.DAT"
240 ON ERROR GOTO 1000:KILL "2.DAT"
250 ON ERROR GOTO 1010:KILL "3.DAT"
260 ON ERROR GOTO 1020:KILL "4.DAT"
270 ON ERROR GOTO 1030:KILL "5.DAT"
280 ON ERROR GOTO 1040:KILL "6.DAT"
290 ON ERROR GOTO 1050:KILL "7.DAT"
300 ON ERROR GOTO 1060:KILL "8.DAT"
310 ON ERROR GOTO 1070:KILL "9.DAT"
320 ON ERROR GOTO 1080:KILL "10.DAT"
330 ON ERROR GOTO 1090:KILL "11.DAT"
340 ON ERROR GOTO 1100:KILL "12.DAT"
350 'SELECTS THE DIRECTORY CHOSEN FROM THE MENU
360 IF A=1 THEN GOTO 500
370 IF A=2 THEN GOTO 540
380 IF A=3 THEN GOTO 580
390 IF A=4 THEN GOTO 620
400 IF A=5 THEN GOTO 660
410 IF A=6 THEN GOTO 700
420 IF A=7 THEN GOTO 740
430 IF A=8 THEN GOTO 780
440 IF A=9 THEN GOTO 820
450 IF A=10 THEN GOTO 860
460 IF A=11 THEN GOTO 900
470 IF A=12 THEN GOTO 940
480 BEEP:LOCATE 23,29:COLOR 7,1:PRINT
      "Incorrect choice":FOR J=1 TO 600:
      NEXT J:COLOR 7,0:GOTO 10
490 'CREATES A DATA FILE THAT CORRESPONDS
      TO THE SELECTED DIRECTORY
500 OPEN "O",1,"1.DAT"
510 WRITE #1,A
520 CLOSE #1
530 SYSTEM
540 OPEN "O",1,"2.DAT"
550 WRITE #1,A
560 CLOSE #1
570 SYSTEM
580 OPEN "O",1,"3.DAT"
590 WRITE #1,A
600 CLOSE #1
610 SYSTEM
620 OPEN "O",1,"4.DAT"
630 WRITE #1,A
640 CLOSE #1
650 SYSTEM
660 OPEN "O",1,"5.DAT"
670 WRITE #1,A
680 CLOSE #1
690 SYSTEM
700 OPEN "O",1,"6.DAT"
710 WRITE #1,A
720 CLOSE #1
730 SYSTEM
740 OPEN "O",1,"7.DAT"
750 WRITE #1,A
760 CLOSE #1
770 SYSTEM
780 OPEN "O",1,"8.DAT"
790 WRITE #1,A
800 CLOSE #1
810 SYSTEM
820 OPEN "O",1,"9.DAT"
830 WRITE #1,A
840 CLOSE #1
850 SYSTEM
860 OPEN "O",1,"10.DAT"
870 WRITE #1,A
880 CLOSE #1
890 SYSTEM
900 OPEN "O",1,"11.DAT"
910 WRITE #1,A
920 CLOSE #1
930 SYSTEM
940 OPEN "O",1,"12.DAT"
950 WRITE #1,A
960 CLOSE #1
970 SYSTEM
980 'CYCLES THE PROGRAM THROUGH ALL
      POSSIBLE PREVIOUSLY CHOSEN DATA FILES
990 RESUME 240
1000 RESUME 250
1010 RESUME 260
1020 RESUME 270
```

```
1030 RESUME 280
1040 RESUME 290
1050 RESUME 300
1060 RESUME 310
1070 RESUME 320
1080 RESUME 330
1090 RESUME 340
1100 RESUME 360
```

AUTOEXEC.BAT

```
DATE
TIME
ECHO OFF
PATH=\BIN
PSC OKI
ASSIGN O:TWO;DOS F:
PATH=\WORD
CLS
ZBASIC DOSMENU
CLS
IF EXIST E:1.DAT GOTO A
IF EXIST E:2.DAT GOTO B
IF EXIST E:3.DAT GOTO C
IF EXIST E:4.DAT GOTO D
IF EXIST E:5.DAT GOTO E
IF EXIST E:6.DAT GOTO F
IF EXIST E:7.DAT GOTO G
IF EXIST E:8.DAT GOTO H
IF EXIST E:9.DAT GOTO I
IF EXIST E:10.DAT GOTO J
IF EXIST E:11.DAT GOTO K
IF EXIST E:12.DAT GOTO L
:A
CD \BIN
DIR /W/P
GOTO Z
:B
CD \UTIL
DIR /W/P
GOTO Z
:C
CD \LOTUS
LOTUS
GOTO Z
:D
CD \WORD
WW
GOTO Z
:E
CD \WORD
CD STEVE
WW
GOTO Z
:F
CD \WORD
CD JANET
WW
GOTO Z
G:
CD \WORD
CD JOHN
WW
GOTO Z
:H
CD \WORD
CD STEPHEN
WW
GOTO Z
:I
F:
CD \GAMES
DIR /W/P
GOTO Z
:J
F:
CD \BASIC
DIR /W/P
GOTO Z
:K
F:
CD \BASIC
CD STEVE
DIR /W/P
GOTO Z
:L
F:
CD \BASIC
CD JOHN
DIR /W/P
:Z
PROMPT Current Directory is $p$_$n$g
```

Vectored from Page 36

885-3020-37 MS-DOS HUG Menu System $20.00

Introduction: The HUG Menu System is a set of programs that allows you to perform all normal computer operations from menus. The menus themselves are generally operated by using the arrow keys to move an indicator to the desired function, and then pressing the RETURN or ENTER key. Using the HUG Menu System, an experienced user can set up disks for inexperienced users that can be used without knowledge of the operating system, etc. The HUG Menu System consists of the menu system itself, a sophisticated menu driven file manager, and a console setup utility.

Requirements: The HUG Menu System requires MS-DOS version 2.0 or higher, and will run on either Z-100 or Z-100-PC (Z-150) series computers. The programs are usable on systems having 128k of memory, but 192k or more is recommended.

The following programs are included on the HUG Menu System disk.

README .DOC
MENU .COM
MENU .DOC
MENU .MNU
MENU .RAW
UTILITY .MNU
UTILITY .RAW
MAKMENU .COM
HFM .COM
HFM .DOC
SETCON .COM
MDISK .DVD
CONFIG .SYS
AUTOEXEC .BAT
MENU .ASM
MAKMENU .ASM
HFM .ASM
SETCON .ASM
MACROS .ASM

Authors: The menu system programs were originally written by Robert Metz, and considerably modified and enhanced by Patrick Swayne, HUG.

MENU.COM -- This is the menu program itself. It uses separately prepared menu files that contain the screen display and commands. It is based on the menu program on HUG disk 885-3008-37, but has been greatly expanded. Using this program, the following operations can be performed from menus:

1. Run a program. Programs can be executed directly by MENU.COM, and batch files are not required as with the menu program on 885-3008-37. Any .EXE or .COM program can be run and any required arguments to the program can be included in the command.

2. Change menus. A sub menu can be set up for selection from the current menu.

3. Change directories. You can change to other main menus in other directories and from them branch to sub menus. In this way, a complex menu system can be built.

4. Change disks. Provision is made to remove one menu disk and insert another. This feature makes possible a menu driven collection of games or other programs on several disks.

5. Utilize COMMAND.COM to perform tasks. Any task that can be performed at the system prompt can be performed from a menu by loading the system command processor (COMMAND.COM) and having it perform the task.

6. Enter manual commands. A menu selection can be set up that will allow the user to temporarily leave the menu system and enter commands manually.

7. Provide on-screen help for each menu item. The user just has to move the indicator to the selected item, and then press the Help key (Z-100) or F1 key (Z-100-PC) to see information on that item.

Setting up a menu is simply a matter of creating a menu definition file using any text editor. You can design what the menu will look like on the screen, and write your own prompts for the keys used. The commands to be executed by MENU.COM and the on-screen help text are placed in the definition file.

MENU.DOC -- This file contains instructions for setting up menus.

MENU.MNU -- This is a sample menu file.

MENU.RAW -- This is a menu definition file, that was used to create MENU.MNU. This file was made using WordStar™.

UTILITY.MNU -- This is a sample sub menu file. A selection on the main menu (MENU.MNU) takes you into this menu.

UTILITY.RAW -- The definition file used to create UTILITY.MNU.

MAKMENU.COM -- This program turns menu definition files into menu files usable by MENU.COM. If you draw boxes around parts of your menu screen display using dashes (--) and bars (|), MAKMENU turns them into graphic line boxes.

HFM.COM -- Move over, Wash, Sweep, and others of that ilk. The HUG File Manager is here! The HUG File Manager is a menu driven file maintenance utility that allows you to copy files, delete files, rename files, create and remove directories, create or change disk labels, type (on the screen) and print files, list files in hexadecimal, and sort or unsort the directory. HFM displays all of the files (or as much as will fit) from the selected directory on the screen and highlights one of them. You can move the indicator (highlighted entry) using the arrow keys to select a file to operate on, or flag several files for multiple copy or delete operations. Commands are executed by moving a second indicator (using Space or Back Space) to select a command and then pressing Return or Enter. You can also execute a command by pressing its first letter. (This method of command selection is modeled after Multiplan™ or Microsoft Word™.)

HFM allows you to page through screens of files if there are too many to fit on the screen, or to specify via wildcards which files will appear on the screen. You can change disks, change the logged drive, and change directories from HFM, and you can copy files from the current directory to any valid path, including non-disk devices.

HFM can provide an alternate method to BACKUP and RESTORE for backing up hard disks. You just flag as many files as will fit on a disk and then copy them all at once, and repeat the procedure until all files are backed up. HFM displays the total number of bytes occupied by flagged files, so you know when you have flagged enough to fill a disk.

HFM.DOC -- Instructions for the HUG File Manager.

SETCON.COM -- This utility allows you to set up certain console parameters on your computer. On a Z-100, you can set the cursor type (line or block, blink or steady), screen wrap or no wrap, key click (on/off), and auto repeat (on/off). On a Z-100-PC, you can set the video mode, the scroll mode, and the cursor type (normal, single line, block).

MDISK.DVD -- The MENU and HFM programs require the MS-DOS command processor, COMMAND.COM, for some functions. These functions will be performed faster if COMMAND.COM is copied to a memory disk. This memory disk is a small one (20k), with just enough room to hold COMMAND.COM without using up too much program memory space. Instructions for patching it to other sizes are included.

CONFIG.SYS -- This file can be copied to your system disks to cause MDISK to be installed, or use it as a guide for modifying your own CONFIG.SYS.

AUTOEXEC.BAT -- This file can be copied to your system disks to cause COMMAND.COM to be copied to the memory disk, and the COMSPEC parameter set up to indicate that COMMAND.COM is on the memory disk. Use this file as a guide to modifying your own AUTOEXEC file if you use one.

MENU.ASM, MAKMENU.ASM, HFM.ASM, SETCON.ASM -- Source code for the HUG Menu System programs.

MACROS.ASM -- Macros required to assemble the above source codes.

TABLE C Rating: (0), (1), (3), (10)

Heath/Zenith
Users'
Group

Vectored from Page 10

```
3040 L=INT(91*RND(91)):IF L<65 GOTO 3040
3050 L$(I)=CHR$(L)
3060 NEXT I
3070 PRINT "Do You Want to Guess the Characters <Y or N>?
     <N>;:CH$=INPUT$(1):IF CH$="Y" OR CH$="y" THEN PRINT
     "—> Good Luck <—":ELSE PRINT "—> I'll Print <--"
3075 FOR T=1 TO 400:NEXT T
3080 IF CH$="Y" OR CH$="y" THEN CH$="Y":C=0:GOTO 360
3085 CH$="N"
3090 PRINT CX$;ED$;
     E$"Y  Practice Char. ----->     out of";N5
3100 PC$="N":GOTO 360
4000 X=10:IF W=1 THEN Y=38 ELSE Y=30
4010 PRINT CX$;SH$;GM$;:FOR T=1 TO W:
     Z=ASC(MID$(L$(K),T,1))-64:Y=Y+1
4020 ON Z GOSUB 5000,5010,5020,5030,5040,5050,5060,5070,
     5080,5090,5100,5110,5120,5130,5140,5150,5160,5170,
     5180,5190,5200,5210,5220,5230,5240,5250
4030 NEXT T
4040 GOTO 380
```

In addition to the above lines, lines 5000 thru 5255 (REMark 42, page 55) must be added. These are the graphic character subroutines I used.

Many thanks to Robert Horn for his program, and to David Warnick (whose articles cause me many hours at the keyboard, long programs, I use HDOS MBASIC not CP/M). And many thanks to HUG for a great magazine.

George L. Motes
6330 Westwood Drive
Rocklin, CA 95677

Hats Off To FBE Research

Dear HUG,

In the November '84 issue of REMark, Mr. Kenneth Strum ("Printing Graphics" article) mentioned that Dave Brockman, head of FBE Research, Inc. (Box 60234, Seattle, WA 98168) had an experimental CP/M screen dump.

I wrote Mr. Brockman to inquire about it and was told that he did not have time to market it, but that I could have it if I sent him a blank disk.

Now I am new to Heath/Zenith, but I know that the rest of the computer people don't do that sort of thing, much less the rest of the world. (Who remembers the Pogue Carburetor? What about the Stirling-engined Ford Torinos?) I was surprised and pleased. FBE research gets my vote for the Nice Company, and they get my business next time, and I suggest others send a dollar their way, as well.

Congratulations for a generous attitude.

Sincerely,

P.G. Manney
1166 Lafayette Road
Lot G-20
Medina, OH 44256

Use At Your Own Risk!

Dear HUG,

In response to the letter from J.T. Malone in the Aug. 1984 issue of REMark, I would like to amplify on the ten mysterious "missing" opcodes. They, in fact, do exist! I have not yet encountered an 8085 on which they don't work. However, NO manufacturer of 8085s will guarantee that they will work. Use them only at your own risk! These opcodes may not be implemented in new versions of the 8085 (i.e. 80C85).

The overflow flag (V) exists only internally to the 8085, and the RSTV instruction is the only one that can act on its contents. V is defined in terms of carry-in (C6) to and carry-out (C7) from bit 7 of the accumulator.

```
V = C7 XOR C6
```

V is affected by all 2's complement arithmetic operations including DAD. For example:

```
  120 = 0111 1000
+ 105 = 0110 1001
-----------------
  -95 = 1110 0001
```

In this example, there is a carry-in to bit 7 (C6), but no carry-out of bit 7 (C7). This causes the sign bit to change when the result should have been positive.

Flag bit X5 is also an overflow flag and is affected only by two opcodes, DCX and INX. When using these two opcodes, if a sign change (bit 15) occurs, the X5 flag is set. JX5 and JNX5 are the only instructions that can act on the contents of this bit.

Yours truly,

Jack Curtis
9750 SW 159th Street
Miami, FL 33157

Will Wonders Never Cease?

Dear Sir:

Please consider the material below for publication as a Letter to the Editor in an upcoming issue of REMark.

In the "Will wonders never cease?" department must come the recent announcement by Peachtree Software, Inc. of long-needed enhancements to their "PeachText 5000" package. I was one of a number of HUGgies who wrote to REMark lamenting the lack of support for common printers and number of other deficiencies. Now, for the princely sum of $10 you can order the PeachText 5000 Toolkit on two disks (by calling 800-554-8900 and giving your charge card number).

Modules are provided to reconfigure your system to support:

1. Printers: Epson FX,MX, Mannesmann-Tally 160, Gemini 15, TI 855, and my Okidata 93.
2. Color monitors
3. List Manager: importing ASCII files, deleting groups of records, and recovering damaged data.

Two new dictionaries containing medical and legal terms, plus the original dictionary words are provided.

Templates are provided for:

1. PeachText: controlling printing in a number of formats (provided as INCLUDE files).
2. PeachCalc: budgets, car expenses, personal calendar, cash, collections, property, and a stock portfolio.
3. List Manager: printing labels, index cards, and merge-printing with PeachText.

These features are long overdue, but are not sufficient to make PeachText 5000 competitive with the second generation of word processors, some of which provide functionality equal to dedicated machines, or to advanced spreadsheets and data bases. If a purchaser can live with its limitations, it will do the job, and it remains a good buy.

Sincerely yours,

Warren S. Hoffman
208 Jackson Boulevard
Deerhurst
Wilmington, DE 19803

In Search Of

Dear Walt,

Please run my short request for assistance, which appears below, in your next issue of REMark.

"I am in search of H/Z89 programs for practicing for college board examinations (like the SAT). I have an H89 running both CP/M or HDOS with 64K and DSDD drives. If you are aware of any sources (be they commercial, private, or HUG), please drop me a line."

Your Assistance will be appreciated, thanks.

Lawrence J. Becker II
29468 Pearson
Selfridge ANG Base, MI 48045

HELP! I Need A Program

Dear HUG,

My system includes the Z-89-11 I/O board (2 serial, 1 parallel). I am using the 8250 DCE port to drive an Okidata 84 printer and the 8255 parallel port to drive a new daisywheel printer. By using the SETLP command, I can toggle the system to output on either one. I have both CP/M versions 2.2.03 and 2.2.04, the latter obtained to recognize and initialize the Z-89-11 chips. I also have a modem (Hayes 300).

My problem is that, apparently, every modem driver program written assumes, if you have an H-89, that the modem I/O will be handled by an 8250 UART. Since both my other ports are in use, I have been waiting for a new program, or a patch to an existing one, to allow me to use the 2661-3 DTE port for my modem. So far, I have been waiting in vain. Neither REMark nor Sextant have contained a solution to my problem.

Does anyone know of a good modem program which uses the DTE port on the H-89-11?

Sincerely,

Charles E. Sohm
903 Bordeaux Avenue
Omaha, NE 68123

Changing your address? Be sure and let us know since the software catalog and REMark are mailed bulk rate and it is not forwarded or returned.

CUT ALONG THIS LINE

HUG MEMBERSHIP RENEWAL FORM

HUG ID Number: ______________________

Check your ID card for your expiration date.

IS THE INFORMATION ON THE REVERSE SIDE CORRECT? IF NOT, FILL IN BELOW.

Name ______________________

Address ______________________

City-State ______________________

Zip ______________________

REMEMBER - ENCLOSE CHECK OR MONEY ORDER

CHECK THE APPROPRIATE BOX AND RETURN TO HUG

	NEW MEMBERSHIP RATES	**RENEWAL RATES**
U.S. DOMESTIC	$20 □	$17 □
FPO/APO & ALL OTHERS*	$35 □	$30 □ U.S. FUNDS

* Membership in France and Belgium is acquired through the local distributor at the prevailing rate.

Heath/Zenith
Users'
Group

Hilltop Road
Saint Joseph, Michigan 49085