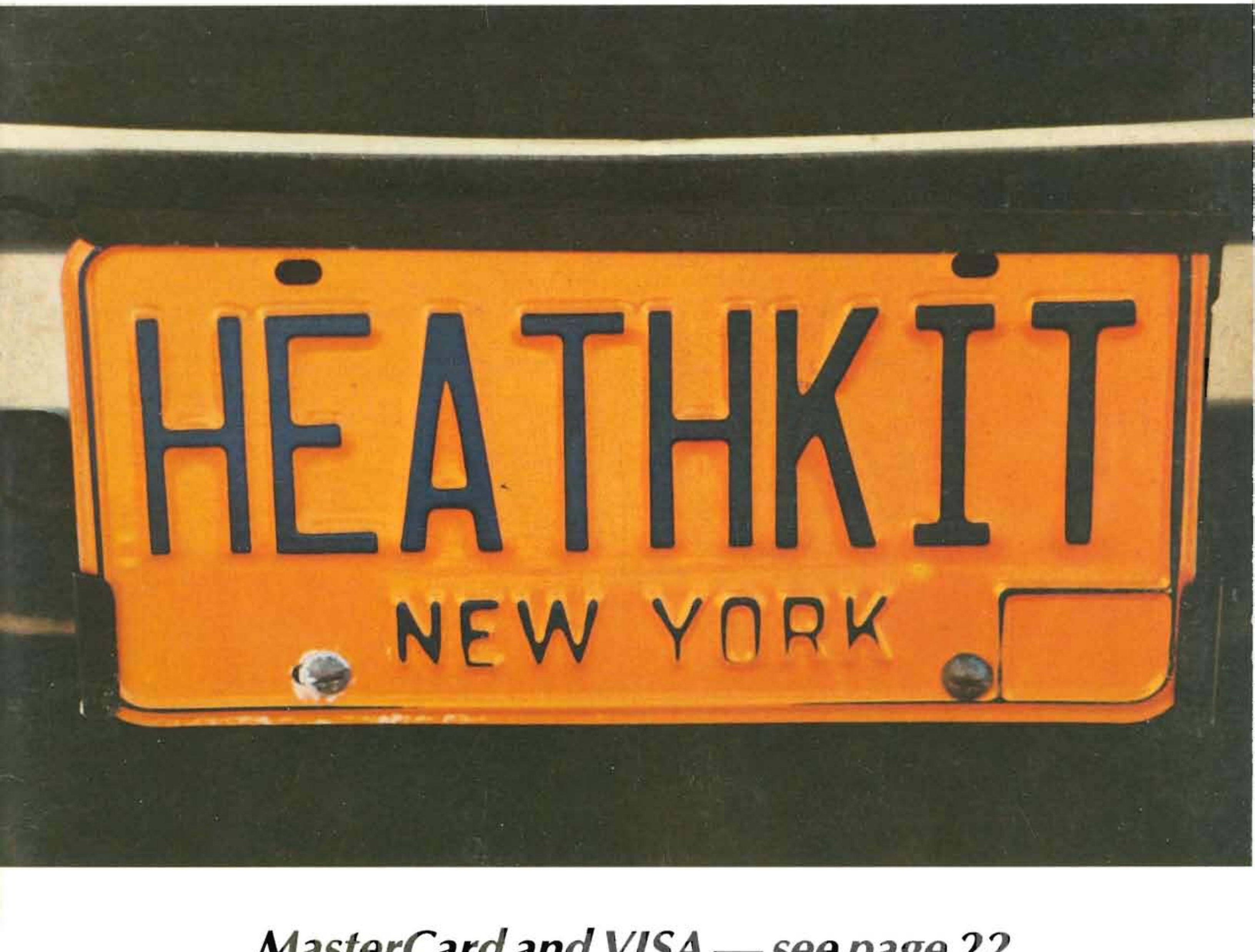
$2.50

REMark

Issue 33 • October 1982

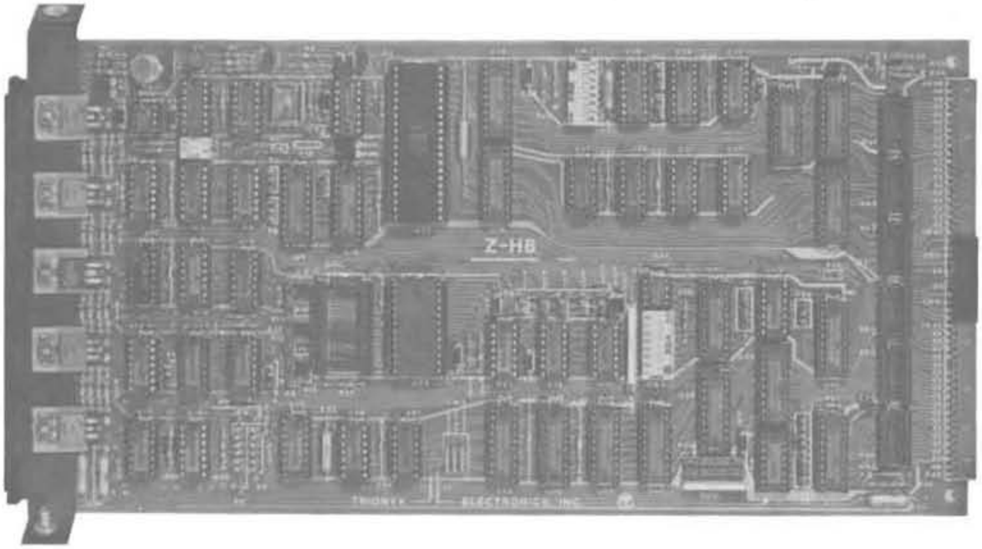


MasterCard and VISA — see page 22

Official magazine for users of HEATH ZENITH computer equipment.

Z80 CPU BOARD for the H8*

* H8 is a Registered Trademark of the Heath Company



Model Z-H8

$ 300.00 Assembled $ 250.00 Kit

Programmable Clock Rate — 2 MHz or 4 MHz Operation

Clock Rate under Software Control — May Be Changed at Any Time

Selection of Wait States (None, 1/2, 1 or 2) at 4 MHz

Buss Termination Network • 51 Integrated Circuits • Transistor Power Supply

Includes Heath Extended Configuration (ORG ZERO) Circuits

Exclusive Z80 Front Panel Monitor ROM — Based Upon Heath XCON-8 ROM

Exclusive Software Speed Utilities (For Both HDOS* and CP/M**) on Diskette

Fully Compatable With All Heath H8* Hardware and Software

* H8 and HDOS are Registered Trademarks of the Heath Company.
** CP/M is a Registered Trademark of Digital Research.

Check • Money Order • VISA • MASTERCARD • C.O.D.
Phone Orders Welcome (714) 830-2092 - Send For Free Brochure

TRIONYX ELECTRONICS, INC.
P.O. BOX 5131, SANTA ANA, CA 92704

REMark Issue 33 • October 1982

on the stack

ON THE COVER: Shown is Rochester NY store manager Jim Milne's,"New York's answer to California's HDOS (appearing in REMark issue 24, Jan., 1982.)

REMark is a HUG membership magazine published 12 times yearly. A subscription cannot be purchased separately without membership. The following rates apply.

	U.S. Domestic	Canada & Mexico	International
Initial	$18	$20	US FUNDS $28
Renewal	$15	$17	US FUNDS $22

Membership in England, France, Germany, Belgium, Holland, Sweden and Switzerland is acquired through the local distributor at the prevailing rate.

Back issues are availble at $2.50 plus 10% handling and shipping. Requests for magazines mailed to foreign countries should specify mailing method and add the appropriate cost.

Send payment to: Heath Users' Group
Hilltop Road, St.
Joseph, MI 49085

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.

HUG Manager .... Bob Ellerton
Software Coordinator
and Developer .... Jim Blake
Software Engineer ...... Pat Swayne
HUG Bulletin Board and
Software Developer .. Terry Jensen
HUG Secretary ..... Margaret Bacon
REMark Editor .... Walt Gillespie
Assistant Editor ...... Nancy Strunk

Articles submitted by users and published in REMark, which describe hadware 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 the software catalog, REMark 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.

KRES™

for Heath/Zenith

Get the expanding world of Kres Engineering working for you.

KRES is the **one** name to remember for TOTAL SYSTEMS SUPPORT

- hardware
- software
- accessories
- technical support

KRES Engineering offers Heath/Zenith users the expertise to solve their computing problems. If you've got problems, call us for their solutions. We've got a catalog full. Or contact us through our **Electronic Bulletin Board** at (714) 559-8579.

Kres Engineering P.O. Box 17328, Irvine, CA 92713 (714) 559-1047 (213) 957-6322

□ **Send me your solutions!**

KRES™
P.O. Box 17328, Irvine, CA 92713
(714) 559-1047
(213) 957-6322

Name:____________________________

Address:__________________________

City:_____________ State:_____ Zip:________

Phone: (_____)______________________

Application: Business________ Personal________

Model #: H/Z-89____ H/Z-89A____ H/Z-90____ H-8____

My specific areas of interest are:____________

H/Z-89, H/Z-89A, H/Z-90 and H-8 are registered trademarks of the Heath/Zenith Corporation.

The Ecology of HUG

The Heath Users' Group is somewhat like a large oak tree. The HUG office being like the main trunk, feeds the various branches, the local HUG groups; and the individual leaves, the members, with materials and support information to help them grow. The leaves and branches in turn give the main trunk life giving juices and its reason to exist, for without branches and leaves no tree can live.

It is through REMark that HUG sets about feeding information to the members. And it is from individual members that REMark gets its material to send out to all the membership so each may grow in their own way in knowledge of Heath/Zenith computers.

To help the flow of information and ideas we have hit upon a new writer incentive program. As in the recent past (since Feb. 1982) if you have a major article accepted and printed in REMark you will receive a one year free membership in HUG. Now, we will also give you your choice of any piece of software available through the Heathkit catalog, FREE. That's right! Any computer software product (1) for Heath/Zenith computers listed in the Heathkit catalog will also be given FREE when you have a major article printed in REMark. As always, smaller articles, Buggin' HUG letters and Questions for the Q & A series are most welcome but do not qualify for the freebees.

What constitutes a major article? Well, your article should consist of no less than 2000 words and be written about software or hardware used with Heath/Zenith micro-computers. Submissions that are written with minimal editing needed and are of general interest to HUG members will be considered. Please submit these articles on disk as text files.

As an additional incentive we will sponsor an "Article of the Year" award. Starting in the December, 1983 issue and each December after we will include a voting card on which members may indicate which article published during that calendar year they think was the best. On tabulating this vote the author of the article voted the best by the HUG membership will be awarded a Heathkit Weather Station (Part #ID-4001) FREE.

This is the opportunity for you to add some great programs to your library (FREE), also a Heathkit Weather Station if your article is voted best of the year and our opportunity to have more varied articles to present to the membership. With this kind of give and take the life blood of our tree can be kept flowing so that all our leaves can have something to feed their creative minds.

Walt Gillespie
REMark Editor

SMASH THE '89's 64K RAM LIMIT!

128K RAM BOARD $595

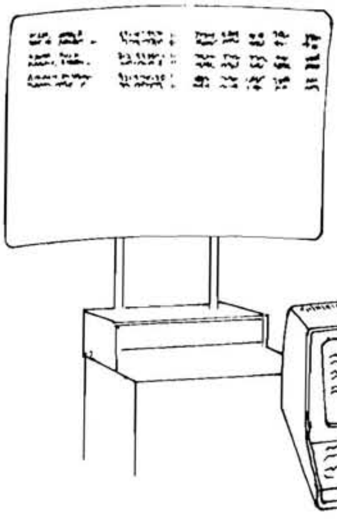
order no 77318

Now have 176K of fast dynamic RAM at your disposal in your Heath/Zenith '88, '89, or '90.

Featuring:

- 128K of 200nSec Dynamic RAM, added to the 48K on your CPU board, for a total of 176K.
- Full compatability is retained for MMS and Zenith CP/M® [64K], as well as HDOS [56K].
- Versatile bank switching technique supports three full MP/M II® compatible banks.
- 112K "Electronic Disk" BIOS module for MMS CP/M included, complete with source code.
- Ultimeth Corp. supplies HDOS support.

Delivery beginning March 15, 1982.



VIDEO OUTPUT $79

order no. 77319

Add an industry standard video output to allow reproduction of the CRT image on another monitor or projection TV system to enhance the usefulness of your terminal in classroom and other educational applications. Allows simultaneous viewing of the display in group situations.

The auxillary display unit should be capable of high resolution for satisfactory performance.

Delivery beginning March 15, 1982.

MAGNOLIA
MICROSYSTEMS

2264-15th Avenue W. • Seattle, WA 98119
(206) 285-7266 (800) 426-2841

LOWER PRICE! 16K RAM $125

order no 77311

Magnolia's 16K Add-On RAM board has been so popular we lowered the price.

DOUBLE DENSITY DISK CONTROLLER $595

order no 77316

Complete hardware and software support for FOUR 8" single or double sided drives and FOUR 5" Single or Double sided, 48tpi [40 track] or 96tpi [80 track] drives, in addition to the three 5" drives supported by your Heath/Zenith controller.

Plus, the obvious advantage of being able to use Single Density 8" media for program and data interchange.

Full compatability is retained with MMS support of the '89s built-in 5" floppy, as well as several Winchester hard disk subsystems.

The package includes:

- Double Density Controller Card.
- Cables for both 5" and 8" disk drives
- CP/M 2.2 on both 5" and 8" media
- New I/O Decoder and Monitor PROMs
- Ultimeth Corp. supplies HDOS support.

If your '89 isn't ORG-0 CP/M compatible yet, our modification is available for $50 additional.

DOUBLE DENSITY SUBSYSTEMS

Dual 8"	DS	48tpi [2.4M]	order no. D8DS	$2695
	SS	48tpi [1.2M]	order no. D8SS	$1995
Single 5"	SS	48tpi [162K]	order no S540SS	$ 945
	DS	48tpi [343K]	order no. S540DS	$1095
	DS	96tpi [700K]	order no S580DS	$1295
Dual 5"	DS	96tpi [1.4M]	order no. D580DS	$1995

COMPLETE SYSTEMS

As well as manufacturing enhancements for the '89 [also '88 and '90], we are a Zenith Data Systems OEM, and have all of their hardware and software products available as well. We can provide a completely integrated system, combining the best Zenith products with our own to provide the exact system capabilities to best satisfy your requirements.

ORDERING INFORMATION

Our products are available from many Heathkit Electronic Centers and independent computer stores throughout the United States. If your local dealer doesn't stock our products, you may order direct or request further information by calling our Sales Department on our toll-free number [800] 426-2841.

CP/M and MP/M II are registered trademarks of Digital Research, Pacific Grove, CA

Buggin' HUG

Dear Pat,

I really enjoyed your article in the 29th issue of REMark, entitled "Improvements to Benton Harbor Basic". Give us more!

In using your "Dollar and Cent" routine on page 8 and 9 of that issue I found an error or possible improvement. It appears that line 10040 should be changed from;

```
10040  FOR I = 0 TO 6-L9:PRINT " ";:NEXT
```

to:

```
10040  FOR I = 0 TO 7-L9:PRINT " ";:NEXT
```

This will make the decimal point location constant for all numbers. Without this change the decimal point will move one place to the right for numbers greater than 9999.99. This results in uneven "right justification" of columns of data.

Sincerly yours,
Vincent Bush
Rt. 1, Box 330
Madison Lake, MN 56063

Dear Walt,

I think a parallel interface such as described in a recent Sextant article by J.C. Hassall for the H89 would be a good idea for the ID4001 Weatherstation. I would be interested in specific information on how to interface the ID4001. Perhaps this would make a good article to publish in REMark and have the side advantage of boosting ID4001 sales.

A.O. Miller
3720 South Drive
Ft. Wayne, IN 46815

Dear HUG,

Just thought I'd drop you a note to let you know about the Votrax TYPE & TALK.

I got one from the Heathkit store and with it also got Tom Jorgensen's device driver TNT. I'm using an H8 with 64K of memory and a Z80 micro so what I did should be OK for the H89-90 too. The device driver works great under HDOS 2.0 but I wanted it for CP/M with MBASIC 5.2. Anyway to make a long

Vectored to 00☞

QUESTIONS & ANSWERS

(EDITOR'S NOTE: If you need answers to specific questions on software or hardware problems please drop us a note, Questions & Answers, Heath Users' Group, Hilltop Road, St. Joseph, MI 49085. Please keep your questions brief and to the point. We will do our best to answer you here in this column in future issues.)

Q. How do I program the function keys on my Z89?

A. The following short program should give you a good starting point on how to use the special function keys in a BASIC program.

```
70  E$=CHR$(27):E1$=E$+"E":
80  PRINT E1$:REM     Clear Screen
90  PRINT "Type A Function     ?    ";:A$=INPUT$(2)
100  PRINT
110  IF A$=E$+"S" THEN PRINT "You pressed f1"
120  IF A$=E$+"T" THEN PRINT "You pressed f2"
130  IF A$=E$+"U" THEN PRINT "You pressed f3"
140  IF A$=E$+"V" THEN PRINT "You pressed f4"
150  IF A$=E$+"W" THEN PRINT "You pressed f5"
160  IF A$=E$+"J" THEN PRINT @
          "You pressed the ERASE key (yes that is a function key)
170  IF A$=E$+"P" THEN PRINT "You pressed BLUE"
180  IF A$=E$+"Q" THEN PRINT "You pressed RED"
190  IF A$=E$+"R" THEN PRINT "You pressed GRAY (I call it WHITE)
200  PRINT
210  GOTO 90
```

Q. How do I get my MX-80 printer to do graphics?

A. The MX-80 printer uses 8 bit characters to do graphics. Therefore, the switches on the printer serial interface must be properly set for 8 bits. CP/M 2.203 handles 8 bit communication to the printer with no problem. HDOS 2.0 requires the LPMX80.DVD driver on the HOS-5-UP disk for 8 bit communication. Benton Harbor BASIC requires that location 12121 which contains the decimal value 127 be poked with the decimal value 255. All other software products will work without modification.

Q. In the Microsoft BASIC programming course there are several example programs that I can not get to work with my Heath/Zenith CP/M BASIC. What am I doing wrong?

A. There are a few minor differences between the HDOS version and the CP/M version of MBASIC. The most common problem is the line continuation in HDOS MBASIC is done by typing the '@' character which is what is shown in the programming course. CP/M users should type the LINE FEED key instead.

Q. When I powered up my H-89 I only heard one beep, and when I tried to BOOT nothing happened. What is wrong?

A. Most likely your OFF LINE key is in the depressed position, or check the cable that connects the CPU and terminal boards, if the problem is not found here then most likely there is a problem with the CPU board itself.

Q. How can I have the underline cursor instead of the block cursor when using MAGIC WAND?

A. To gain the underline cursor in MAGIC WAND as opposed to the block cursor patch address 27E from 78 to 79 as follows:

Using DDT (Dynamic Debugging Tool)

```
DDT EDIT.COM
R
S 27E
79
↑C
SAVE 16EDIT.COM
```

story short I'm using the TTY device driver in CP/M 2.2.03. Using CONFIGUR.COM I changed the LST: device from LPT: to TTY: and the baud rate to 4800 baud to keep in line with my H14. I'm also using the WHA-13(?) cable for the TYPE & TALK, anyway using that combination I can now use the PIP at command level to output to the Vortrax or in MBASIC use the LPRINT or LLIST function to do the same thing.

Hope this can help people in the same boat I was in.

Happy Hacking,
Bruce Bennett
574 Printy Ave.
Milpitas, CA 95035

Dear Walt,

Please don't discontinue the articles by Henry Fale on the Lucidata Pascal.

Last year I got very interested in Fortran, Fortran Corner REMark Issues 12 and 13. After two issues the articles were discontinued.

I have purchased the Lucidata Pascal and the update. The articles are a great help in understanding the Pascal Language and the odds and ends that are necessary to make this language run on the Heath computer.

So I ask again please don't discontinue the articles by Henry Fale.

Thank You,
Donald A. Nicholls
8085 Amity Pike N.E.
Plain City, OH

(ED: Don; Like most writers Henry is a contributor, and as such we have little control over when articles are submitted. I just talked to Henry and he thinks he will shortly have finished what will be his final article on Pascal. When it is received it will be included in the following issue of REMark. Because of increased business pressures Henry has found it harder to find time to write.)

Dear HUG,

The UCSD PASCAL System Users' Society (USUS) will hold its semi- annual national meeting in Dallas, Texas, October 29-31.

The meeting, which follows ACM '82, will be held at the Harvey House Hotel in Dallas. Keynote speaker for the meeting will be John D. Page, vice president of engineering for Software Publishing Corp. (Mountain View, CA).

For further information contact:

A. Winsor Brown
USUS Vice President
Huntington Beach, CA
(714) 891-6043

Dear Nancy;

I am attempting to organize a Heath Users' Group in the Nashville, Tennessee area, and would welcome contacts from prospective members.

I am available after normal business hours at the following phone number: (615) 361-4892.

The new format for REMark is very attractive, and adds a polished look that has been lacking. You should put titles on the article continuations in the back pages. If two articles on a page are 'vectored' to the same page, it can be a little confusing figuring out which piece goes with which article (c.f. REMark #31, page 8 vectored to page 30).

I really enjoyed the first National Heath Users' Group meeting in Chicago. I expected to be attending the next one.

Keep up the good work!

Sincerly,
Charles S. Stewart
3144 Country Meadow
Antioch, TN 37012

Dear HUG,

In the July issue of Dr. Dobbs, Arne Rohde discusses a method of resetting an MX80 by adding a switch that pulls the INIT pin of the parallel input low. Clearly, this will work on those systems using the parallel interface. Those of us who are using a serial interface with a 2K buffer will have to take additional steps to make the reset work. My approach is to use a double pole momentary contact switch that pulls Pin 4 (Reset) of the 8048 microprocessor low at the same time as the INIT pin. This is best accomplished by wiring one pole of the switch across C3 on the back (trace) side of the serial interface board. The other pole connects from R15 near IC-3-E and the anode of ZD-5 (ground) next to R15. Use caution when soldering the connections, and inspect them for shorts, etc. before powering up the printer. This modification will allow you to reset the printer without turning off the power as was required before. It will work on MX80 and MX100 printers as well.

Here is an interesting discovery I have made about the MX100, if you set S1-3 'on' and S1-4 'off' the MX100 will power-up in a mode that will print 12 (count-em) characters per inch. This only seems to work at power-up or reset and cannot be selected by software means. All software commands seem to do exactly what they are supposed to do. Of course this will probably not work for Graftrax Plus because the switch settings have completely different meanings.

David E. Myhr
1461 Tyler Park Way
Mountain View, CA 94040

"Actually it's very simple! Just insert the diskette, press B for BOOT, and hit a RETURN."
(ED: The young gentleman pictured above is Bob Kalman, one and a half year old son of Robert Kalman of Zenith Plant 40. I am told that young Bob prefers a computer to a toy truck for something to occupy his leasure time.)

AUTOFILE Update

(P/N 885-1110)

Jim Tysinger
118 Shannon Hts. Dr.
Verona, PA 15147

Editor's Note: *This letter by Jim Tysinger was the best way to publish the changes to the old version, plus introduce version 2.0.*

Dear Terry,

Enclosed are two disks containing a new version of AUTOFILE, pertinent documentation and source for all .ABS modules as we discussed in our recent telephone conversation. I refer to this revision as Version 2.0 because of the new functions implemented. These functions allow the user to:

1) Unload a data base into an editable text file,

2) Specify source for the "STORE ITEMS" function as a file on disk,

3) Produce a printout of the keywords used in a data base along with the occurrence count for each keyword.

I have found these functions useful in my data base activities, so they were incorporated into the existing AUTOFILE menus in appropriate places. The first two functions facilitate data base maintenance and the third has proven valuable in selecting keyword association and search strategies. A new LP: device handling feature allows users with no LP: device the limited use of a disk file substitute for the text normally sent to the printer, or the ability to disable print functions completely. All documentation has been revised to reflect these changes.

All of the operational documentation is on the absolute binary disk, along with a step-by-step example for new users and the PRGINDEX data base illustrating the use of AUTOFILE for a catalog of programs. The source for this data base was the program index in issue 24 of REMark. A new user will probably feel a little more comfortable with the additional documentation and data base.

The source disk contains a documentation file specifying the program used to assemble the .ASM files on the source disk to produce the corresponding .ABS files on the absolute binary disk. All .ABS files can be reproduced exactly from the source supplied. The procedure for combining two of the .ABS files into the resident AFL module is also described. The mnemonics used now correspond to the Z-80 set reproduced in my copy of Heath manual #595-2268-02, which should eliminate some user confusion regarding interpretation of the instructions (if their manual is the same as mine).

There is some space left on each disk for the addition of your "README.DOC" file. If you choose to distribute Version 2.0, I recommend that you include all files on both disks, as there are references to them in the documentation, and they are all useful in understanding the program operation.

Two problems have been discovered in the old AUTOFILE code. These are:

a) When performing a subject or a display field search, items having numbers above 127 are not found. This problem does not occur when performing a keyword search.

b) The data base re-organization function does not write out a complete new keyword-inverted file if there is an item number 127 with keywords in the data base.

These problems can be fixed with the following modifications using PATCH (notation is PATCH-oriented, i.e. old/new):

a) PATCH AFL.ABS

Address	Value
111121	062/006
	163/052
	111/117
111161	375/076
	167/000
	000/002

b) PATCH AFUTL.ABS

Address	Value
53115	050/312
	043/075
	124/057
	135/124
	257/135
	001/345
	124/257
	001/001
	355/124
	261/001
	345/355
	142/261
	153/343

Address	Value
53170	001/076
	377/377
	177/013
	043/043
	176/276
	271/312
	050/172
	373/053
	022/355
	023/240
	270/352
	040/174
	366/053

Address	Value
57071	122/111
	111/106
	115/056
	101/212
	122/052
	131/000
	040/043
	111/021
	124/062
	105/112
	115/267
	040/355
	106/122
	111/104
	114/115
	105/303
	056/162
	212/053

These are lengthy patches, but some code had to be moved to allow for an insertion in AFUTL.ABS.

I applied the above fixes using PATCH, and had no problem with the procedure, even though the only PATCH documentation I am aware of is in REMark issue 7. The user should verify the old values listed, just to be sure they are going in the right places, as there is no verification capability in PATCH.

You can use your judgement regarding the problems discovered and the fixes, but I thought you would like to know about them. I am not aware of any other problems in the old version, and these have been fixed in Version 2.0.

Sincerely,

Jim

BASIC

Another Stab at Real-Time Input
for MBASIC and Other languages Using the Type-Ahead Buffer in HDOS

Dale Grover
3201 Chicago, Ave.
Stevensville, MI 49127

Real-time input for MBASIC allows the program to check and see if the user has typed anything, without "hanging up" or waiting for the user to type something. Arcade type games are real-time — they don't wait with frozen screens while the player decides which way to jiggle the knobs or push a button. There are several different approaches to real-time input that have been described in REMark; I propose a method that is perhaps less error-prone and more understandable. And, it worked. (Never say "works"—no program ever "works". It may have worked in the past, it may work again in the future, but as for in between,. . .)

One method of real-time input for MBASIC involves the use of a machine language subroutine residing along with MBASIC and the user's program in memory. Issue 18 of REMark on page 24 contained an article on such a routine. In this instance, the machine language routine was not POKE'd into memory (which must be "safe" from MBASIC and the user's program and arrays), but was placed into a numeric array as data, then called by using the VARPTR function to generate the starting address. This routine, when disassembled, is

```
MVI      M,0     place a 0 into the "output memory location
                 (this is the FAC—floating point accumulator)
SCALL    .SCIN    go get a char, if one is ready
                 (actually, this is the sequence
                 RST      7,DB      1
                 and calls a routine in the monitor)
RC               return if no char has been typed
MOV       M,A    put char (which has been returned
                 in register "A"—not MBASIC variable
                 "A") into "output" memory location

RET              return (to program)
NOP              no operation (here only to fill up data)
```

The actual machine code version of the routine is the sequence of numbers (in HEX):

```
36 0 FF 1 D8 77 C9 0
```

Because MBASIC uses two bytes to store each INT type element of an array, the routine must be broken down into groups of two numbers each and "reversed". Hex is very handy for this, as two hex digits equals on 8-bit byte (one PEEK or POKE location in memory). So now, the routine is

```
36 1FF 77D8 C9
```

Along with the USR function, this routine is used to generate the real-time input in many of HUG's MBASIC games. The actual form of the statement in the MBASIC program, once the routine has been placed in memory and the USR has been set, is

var = USR[number](dummy variable)

For example, the following is from the demonstration program:

X=USRO(0).

However, the added lines to a program using this routine (and their complexity) and the ever present danger of getting the routine messed up (say, by accidently changing the value of one of the variables) suggests there has to be an easier way.

Another method for real-time input uses the type-ahead buffer kept by HDOS. This buffer has been described in REMark issues 11,14 and 19; the latter in more detail and with a BASIC program to place information INTO the buffer, rather than extract data typed in.

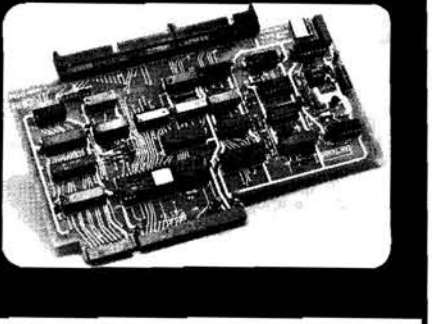
The type-ahead buffer is used to store data typed in at the terminal until it is actually called for by a program. When MBASIC executes the statement;

INPUT G$

for example, the buffer is the place that is examined for information, not the keyboard. Any data typed in before the statement (and which had not been called for by MBASIC—looking for a command line, perhaps—or the user's program) would be returned just though it had been typed in at that moment.

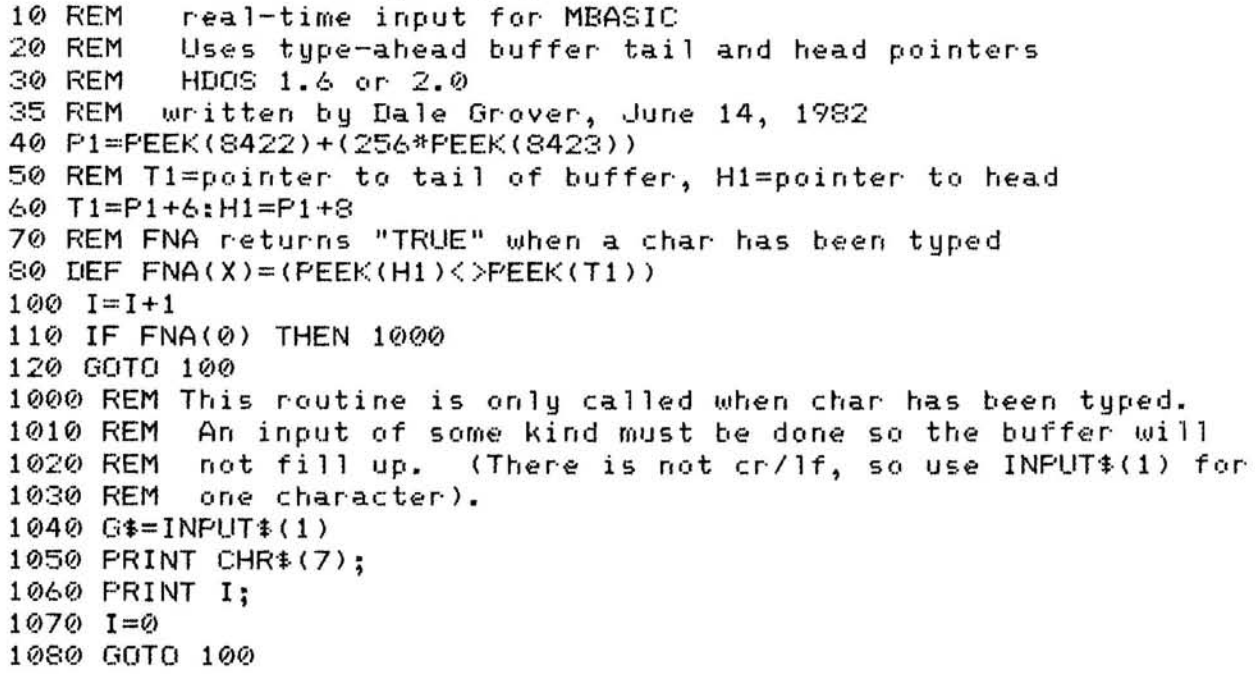
This is the fact that can allow an MBASIC program to see if anything has been typed in, so that an INPUT$(1) is only done when there is already something to be had, rather than possibly having to wait for the user to type something in.

HDOS keeps pointers to the beginning and end of the type-ahead buffer which are readily accessible in MBASIC. The memory loca-



tion 8422 (and 8423, which is the hi-order byte) contains a kind of "base" address in HDOS. To find the address of the head pointer, add 8 to the base address. The address of the tail pointer is found by adding 6 to the base address. By PEEKing these two locations, the head and tail pointers, it is possible to determine if there is any information being stored in the buffer. When both pointers are the same, there is nothing in the buffer. Only when they are different can an INPUT$(1) function be carried out without stopping the program.

The program below illustrates the basic principle. The user- defined function FNA is optional—all that needs to be done is to check the values (by peeking) of the locations whose addresses are in H1 and T1 before doing an INPUT$(1). In this particular program, the real-time aspect of the input is reflected in the number displayed on the screen, which is the number of "tics" (increments of I) since the last input.



```
10 REM   real-time input for MBASIC
20 REM   Uses type-ahead buffer tail and head pointers
30 REM   HDOS 1.6 or 2.0
35 REM  written by Dale Grover, June 14, 1982
40 P1=PEEK(8422)+(256*PEEK(8423))
50 REM T1=pointer to tail of buffer, H1=pointer to head
60 T1=P1+6:H1=P1+8
70 REM FNA returns "TRUE" when a char has been typed
80 DEF FNA(X)=(PEEK(H1)<>PEEK(T1))
100 I=I+1
110 IF FNA(0) THEN 1000
120 GOTO 100
1000 REM This routine is only called when char has been typed.
1010 REM  An input of some kind must be done so the buffer will
1020 REM  not fill up.  (There is not cr/lf, so use INPUT$(1) for
1030 REM  one character).
1040 G$=INPUT$(1)
1050 PRINT CHR$(7);
1060 PRINT I;
1070 I=0
1080 GOTO 100
```

MAKE YOUR OWN 5 1/4" DISK FILE BOX

Gary Jewell
Capital Heath Users' Group, Inc.

EDITORS NOTE: The following article was taken from the June Issue of the >CHUG Newsletter, Capital Heath Users' Group, Inc., P.O. Box 2653, Fairfax, VA 22031. We thought it was a good idea and wanted to share it with you.

I often thought I would like to have a disk file box for 50 or so of my 5.25" disks but was insulted by the $20-40 price for them. Being basically cheap, I figured I could make one less expensively and maybe even better. I found the perfect containers in a department store and the rest was child's play. When completed, these become clear plastic boxes, hinged at the rear and large enough to hold about 70 mini-diskettes.

1	"SUPERSEAL CLEARS" 48 oz. Flat Food Saver .	$3.00*
1	"SUPERSEAL CLEARS" 82 oz. Flat Food Saver .	4.00*
1	pkg. of 4 Brass Hinges 3/4"x1" (only need 2) .	1.40
8	round-head 6-32 x 3/8" machine screws . . .	.24
8	6-32 machine nuts	.24
	TOTAL COST	$8.88

** These are approximate prices at Giant, Hechinger, Sears, etc. I have gotten both these items for $1.99 on sale at Zayre.*

1. Remove the lids from the food containers. (If someone comes up with a use for these please let me know — I have lots!)

2. Notice a lip or ridge running around the top of the containers. This lip must be removed along one narrow side. I use a bench grinder and it takes about 2 or 3 minutes to do each one. Be careful here, they are easy to break at this time. Do both containers.

3. Place the two containers on a flat surface with the "lip- less" side up, mating the openings to each other. (I use small wedges under the boxes to offset for the ridges on the bottom.)

4. Now, place the hinges at the obvious points and mark for the 8 holes to attach them (a nice sharp scratch awl works great).

5. To drill the holes, I use a piece of scrap 2x4 jutting out from the corner of my workbench. This supports the container well.

6. Fasten the hinges on the outside of the box with the screws coming from the inside (round screw-heads on the inside). Don't overtighten the nuts or you'll break the plastic.

7. A piece of foam (like the kind Heath packs their circuit boards in) on the bottom accounts for the curve there and cushions the disks. Rubber feet on the bottom will keep it from sliding around.

I've made several of these and they not only store my disks, but they make nice presents for other computer folks. I have thought of using epoxy glue instead of screws or even plastic hinges with plastic cement. If someone tries these, let me know if they work and are strong enough as either way would be lots easier. I have also tried "pop-rivets" instead of screws—broke the boxes.

WORDSTAR, Reader Review

Elliott S. Kanter
1387 St. James Street
Benton Harbor, MI 49022

Sooner or later, the average computer user will tire of playing video games, writing programs, or for that matter fooling around with data bases, and want to get down to some serious applications. Perhaps, I'm a bit frivilous, but, for me a serious application for my computer is the generation of text, words, articles and books. A computer is a natural for doing all of the secretarial type functions, in fact, with a program designed to process words [a Word Processor], your computer becomes an automatic electronic typewriter, with some real advantages over that 'beat-up' typewriter you have sitting on a shelf.

Obviously, when you create or write something, it appears on a medium, usually a page of paper, you review it, look for errors, and then either use some correction fluid, or retype the entire page, time and time again, until you are satisfied with the results, or the page is "letter-perfect". There's something about the retyping that crimps the creative juices and makes for what some people have called "writer's block". Well, with your computer, the CRT becomes the page, your disk drives and the attendant circuitry become a temporary "scratch-pad" to hold your thoughts, and never again are you forced to retype for the sake of retyping, or for that matter to ensure error-free copy.

This article is being prepared on a most versatile and easy to use word processing system called WordStar. It is available for most disk configurations supported by the Heath/Zenith Family of computers from the single sided, single density hard-sector, through the 8-inch floppy systems. However, unlike other word- processing systems in the Heath/Zenith Catalog, WordStar *requires* that you have more than one disk drive and at least 48 K bytes of RAM available. Obviously, when you need more than one disk drive to support a system, there must be something really special about the software. In the case of WordStar, that would be a definite yes! Of course, there are the usual bells, whistles, and features that you would expect to find in a word or text processing system, but, WordStar, offers an added something, ease-of-operation!

Having tried most of the commercially available word-processing software compatible with the H/Z-89, I can safely state without fear of contradiction, that WordStar is by far the easiest system to use. "Easy" is a term which could have one meaning to me and yet another to you, so let's look at a representative example, as it were, the complete novice computer user — my wife. Deb who can use an electric typewriter, thought the computer and the room it took up in her sewing room was a waste and what did we need it for. Sound typical, no doubt many of the married computer-users who read REMark have heard similar comments. I brought home the manual as well as the disks and decided to convert her to a bonifide wordprocessing/computer hacker.

Obviously, you don't hit her up front with a manual to read, but, even if I had, she would have read that "....the best way to learn WordStar...is to go ahead and use it.." I elected to take that approach, with a simple list of guidelines which are repeated for those with similar problems:

1. Insert the disk with the labels both facing the "opened door". Close the door (here we assume the power is already on)

2. Press the "B" key, then press the return key.

3. When that A> appears on the screen, type WS and then press the return key.

4. As soon as you see the WordStar menu, you are ready to "create" your masterpiece.

Essentially,that is about all you must know initially to begin using WordStar. Of course, the disk had already been SYSGEN'ed with CP/M, and in our case, it was more or less a 'data-disk' because we use a dual density, double sided 96 t.p.i. drive, but in essence, that is all you have to do.

Unlike other word-processors, WordStar tends to both walk and talk you through the creative process. We have twice used the term "create" and for good reason. With a word-processor,you *create* a file. The file, is your document. Thus, if you consider each of your 'works' a *creation*, you are well on your way to learning exactly how to use WordStar. In one way, we have, however, placed the cart before the horse. WordStar does require that you run the INSTALL program before you initially use the processor. What install does is to configure the word processing program to your particular operating environment, which is an "arch" computer-type buzz word for the kind of system you own and more specifically, what sort of a printer you use. By the way, if you haven't got a printer, then the best features of WordStar will be lost. So if you haven't got a printer already, stop what you are doing, and get one. A good choice might just be the Heath H-25, a high speed dot matrix printer that can print a full typewritten page in about thirty seconds, give or take a moment. INSTALL requires that you answer a few questions in response to prompts [another computer- buzz- word meaning questions], after you have answered these prompts, your system will be configured, or set up for your particular operating /computer/printer system.

We've covered a few of the key steps in getting started, we can safely assume that you have a computer, more than one disk drive, and have run the INSTALL program to tell your processor just what your computer's world is all about. Now, a slight deviation from what seemed like a hard fast statement. It is *possible* to run WordStar with *only* one disk drive. You will have to prepare disks with the WordStar overlays, messages, and then SYSGEN them to ensure that they are bootable. Here we go again, another computer buzz-word. **Bootable** means that the disk can be loaded into the computer without first loading CP/M. There is a price you will have to pay, though. The disk does not have unlimited storage capabilities, and the "system" takes up a fair amount of the available space, *before* you begin to create. So, you will probably not be able to get more than five pages on a disk that also contains the WordStar system, and is sysgen'ed. It seems that every page of text takes up about 2000 Bytes (2K bytes) of available storage memory on the disk, and the 5.25 inch hard sector disk has only about 160 K Bytes to start with. Let's take about 50 or so "K" away to allow for 'overhead' and what we have left will allow you about a safe 5 or ten pages of text allowing for 100% backup which WordStar can accomplish with little or no intervention.

It's time to take this one step further and assume again that we have our system up, running and we have more than one drive. In fact, the system (WordStar, etc) is at home in drive A and we have placed an ordinary *formatted* and otherwise empty diskette in drive B. When we call up our WordStar, we have a variety of choices to make, but, let's make this simple, after all, WordStar is simple and select "L", either upper or lower case. According to the menu, we have just elected to change the logged drive, we are told that the current drive is A, which corresponds to the location our system [operating WordStar etc.] is residing in, and we are asked to tell the computer what drive we wish to use, add a colon and a return.

If we type B:<return>, there will be some whirring sounds and the screen will now display the new logged drive B together with a list of the files (there won't be any if the diskette is blank). We have just changed the logged drive and have about 160 K bytes available for our first creation. According to the menu, we should enter D for create a file, the computer will then ask for a file name, we have up to eight letters, some numbers and or a combination of both to name our file. If we select a name which is too long, WordStar will edit the name and display it in the topmost area of the screen as B:CREATION. You don't have to be concerned with upper or lower case, because the file name simply tells the computer what you want to call your creation. You could call your first attempt "Attempt" or whatever strikes your fancy. The computer will now acknowledge that it is a new file and ask you to wait by printing the letters WAIT! It's really quite an intelligent program,or you are rapidly becoming used to a computer.

After a few seconds you will see a new screen with a menu and data line which tells you what drive, what the file's name is and the current position of the cursor by row and column. It also tells you you are on page one, and you will see a ruler which gives you an indication of the line width and tab settings. Now all you have to do is type your creation, or if you want to be 'computer-literate', input the data. You will notice that you do not have to enter a carriage return. WordStar has a normally enabled (turned on) function called word wrap around which continues the sentence to the next line. In fact, the only time you enter a carriage return (a really dumb term, there is no carriage on a computer) is to end a paragraph. All of the other keys operate the same way they would on a typewriter, except, the CAPS LOCK, which will give you all capital letters [uppercase], but *has no effect on the numbers*, they will still remain and print as numbers rather than the other or shifted characters they can print if you use *either* shift key.

If you have been trying this right along with the text, you may have already filled up a page or so of text and noticed how even the right hand margin is. This is because the WordStar system will automatically right-justify text. This means that it will add spaces where necessary to permit the right margin to always be even. Perhaps, you have even gotten to the point that you'd like to try some special effects, like bold facing words, if you press CTRL [the control key] and the letter P, you will see a new menu appear which tells you that CTRL PB will begin the bold facing and repeating the CTRL PB ends the bold face operation. This looks like this on your page, the next ↑ B word ↑ B is the only **word** which will appear in bold face. The P is silent and doesn't appear. Despite what may look like a ragged margin on the right, the final printed version will still be even and neat. These control characters do not print on your paper. Let's try another, how about underlining [computers call this underscoring]. Press CTRL PS to start and CTRL PS to end, thus, if we wanted only the next ↑ PS word ↑ PS to be <u>underline</u>, thats the way it would appear on the screen. Again the P is "silent", and the printed page will appear with the margins even. You didn't see it, but I paused a few lines back and "saved" what I'd written thus far to disk. You might want to save every page or so, just in case your two year old comes in and crashes the system, or the house catches fire or some other calamity. This save procedure will protect what you have written and store it on the disk. To save a portion of the document, pres CTRL KS which tells your computer to save what has been written to disk and tell you if you want to return to the point at which you left the 'creative process', just press CRTL QP before anything else. If, however, you had finished what you were doing you would type CTRL KD and tell the computer I've finished, save the copy on disk for me. There are two other commands you could use, KX, which saves the data and returns to the system prompt (in this caseB>) or KQ which says, I was a dummy, and don't want this to appear in a file, dump the whole thing.

Considering that you started reading this article not knowing much if anything about a word-processor, or more specifically WordStar, we've come a long way, you correct your errors by using the DELETE key with the cursor positioned one character to the *right* of the character you want to delete. For longer errors, the menu tells you that a CTRL G will delete a character, CTRL T will delete the word to the right or CTRL Y can delete the entire line. Not bad, because you already have mastered probably all the goodies you will need for normal correspondence, that is unless you really want to be fancy and print letterhead with each line centered. Then all you have to do is type the information, use the cursor movement commands CTRL E, and X and the command CTRL OC to center the entire line.

Why is this system so easy to use, or what have I left out of this review. The answer is, it is easy to use, because it has certain predetermined conditions called "defaults". These defaults determine the line spacing (single spaced), the size of the paper, its heading margins, left and right margins, and the fact that the right-justify is on. If you don't want these conditions, then you read your manual and change them, you didn't think this review was going to do everything for you——did you?

It probably took you about ten minutes to read this review, and you have already learned nearly all that you will need to know about using WordStar, which in my book makes it quite an easy system to learn. One last piece of information, to print your file, after you have saved the 'creation' enter P for Print, and if you are the sort that accepts these defaults, you can then press the ESC (escape) key, that is assuming your printer is connected and on-line, if you are curious, you could elect to press the return about six times and watch all of the default conditions appear, either way, you will have printed your first WordStar document.

If you have any questions that I might not have answered in this review, drop me a line, please enclose a self-addressed stamped envelope if you have a question, if you are as happy as I am with WordStar, then just tell me and all of your friends. Stop into your nearest Heath Zenith Computer Center and ask for a demonstration if you are still skeptical. That's it, and my next key strokes are CTRL KD.

MBASIC 4.7 TRIG FUNCTION ERRORS

Alan Bose, President
Taildragger Flyers, Inc.
2514 Essex Court
St. Joseph, MI 49085

If you are using the trig functions in Microsoft BASIC version 4.7 you should be aware of the following bug. The problem occurs when the argument in the SIN or TAN function approaches zero. In this situation MBASIC should return the argument no matter how small the number. Instead MBASIC 4.7 returns a zero.

Running the following short program allows you to compare how MBASIC treats the SIN, TAN & ATN functions of progressively smaller numbers. The problems with SIN & TAN will be self-evident.

```
 10 'SINTAN.BAS
 20 'TEST MBASIC SIN, TAN & ATN FUNCTIONS
 30 X=1E-4/3
 40 PRINT CHR$(27);"E"
 50 PRINT "I","X","SIN(X)", "TAN(X)","ATN(X)"
 60 PRINT
 70 FOR I=1 TO 36
 80 X=X/10
 90 PRINT I,X,SIN(X),TAN(X),ATN(X)
100 NEXT I
110 END
```

When you run the program you will see that SIN & TAN begin returning zero on the third cycle. The ATN continues to return the argument through the 34th cycle and after that even X is too small to be represented.

Run the same program on MBASIC 4.82 or 5.21 (the CP/M version) and you will see that the bug has been fixed. All three functions will continue to return the argument through the 34th cycle.

It is possible to program around the bug by implementing the following short subroutines:

```
10000 'SIN SUBROUTINE      ARGUMENT: X
10010 IF ABS(X)>0.000001 THEN X=SIN(X)
10020 RETURN

20000 'TAN SUBROUTINE      AGRUMENT: X
20010 IF ABS(X)>0.000001 THEN X=TAN(X)
20020 RETURN
```

The subroutines seem to work fine as long as you don't have to call them from a user define function. Of course, for general use version 4.7 works just fine. But if you're using trig functions and an unexpected result creeps up, this just might be the reason.

Recovering Deleted Files In HDOS and CP/M

Pat Swayne
Software Engineer

It can happen to anyone. You're happily working (or playing) at your computer, and suddenly you discover that you have deleted your only copy of that new program you have been working on for 5 hours. Well, all is not lost if you have a good disk patch utility (DUMP program), and if you have not written anything to the disk since you deleted the file. If you have no other reason for having such a utility, this is a good one. HUG offers DUMP (hard sector disks only) on disk 885-1062 and UDUMP (all disk formats) on 885-8004 for HDOS and SDUMP on 885-1213 for CP/M. HUG also offers the programs UNDELETE and SAVE for recovering deleted HDOS files, on disk 885-1120. See the New Software section of this issue for a description of those programs.

RECOVERING DELETED HDOS FILES

In REMark issue #19, we presented a method for recovering deleted HDOS files, but it is a rather complex procedure. HUG member Larry T. Wier developed a new method, which is presented here.

You will need at least one initialized disk in addition to the one containing the deleted file. The second disk can be your system disk, but it must have as much free space on it as the file you deleted. Your system disk should also have DUMP or UDUMP on it. Now, follow these steps.

1. Locate the deleted file in the directory file, DIRECT.SYS. If you are using DUMP, you can use it in the file mode to locate the track and sector of DIRECT.SYS on the disk with the deleted file as in this example:

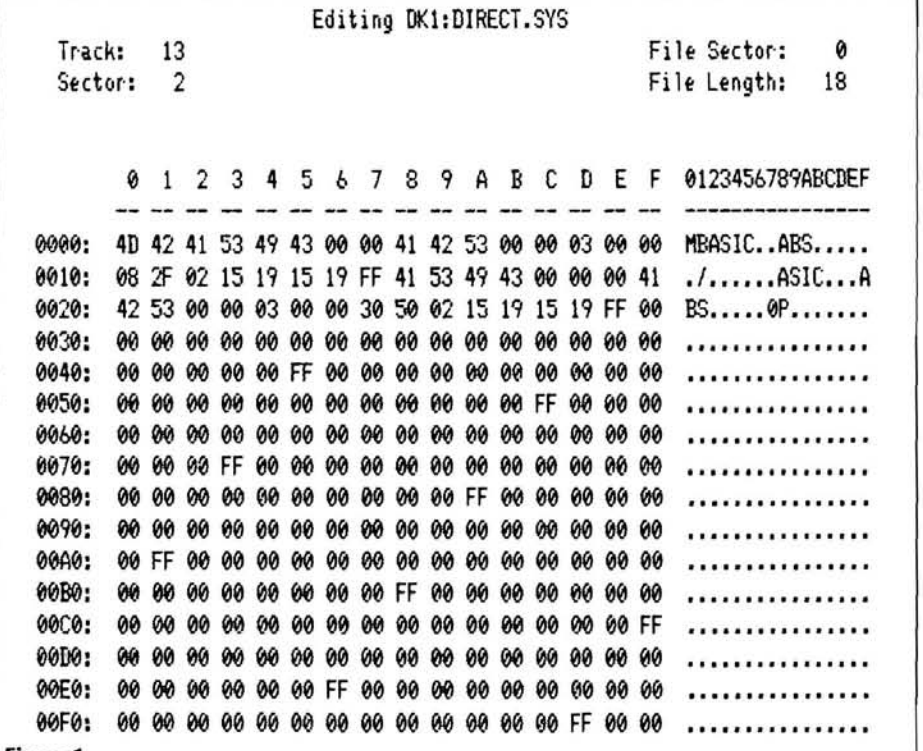
```
>DUMP SY1:DIRECT.SYS
```

The disk with the deleted file is mounted in SY1: in the above example. Type Control-C as soon as DUMP starts listing lines on your terminal. Then note the track and sector displayed at the beginning of the dump listing. This is where to start looking for the deleted file. Run DUMP in the track-sector mode (just enter DUMP followed by a return) and enter the starting track and sector of DIRECT.SYS when it prompts for them.

If you have only one drive, you will not be able to use DUMP in the file mode to locate DIRECT.SYS unless the disk containing the deleted file is a system disk and already has DUMP on it. You will have to run DUMP, remove your system disk, insert the deleted file disk, and start searching at the default location of DIRECT.SYS, which is track 13 sector 2 for HDOS 2.0, and track 22 sector 2 for HDOS 1.6 on standard 5-inch hard sector disks.

If you are using UDUMP, just use it in the file edit mode, and enter DIRECT.SYS (preceded by a drive designation, if necessary) as the file to edit. If you have only one drive, you can use UDUMP to dismount your system disk and mount the disk with the deleted file.

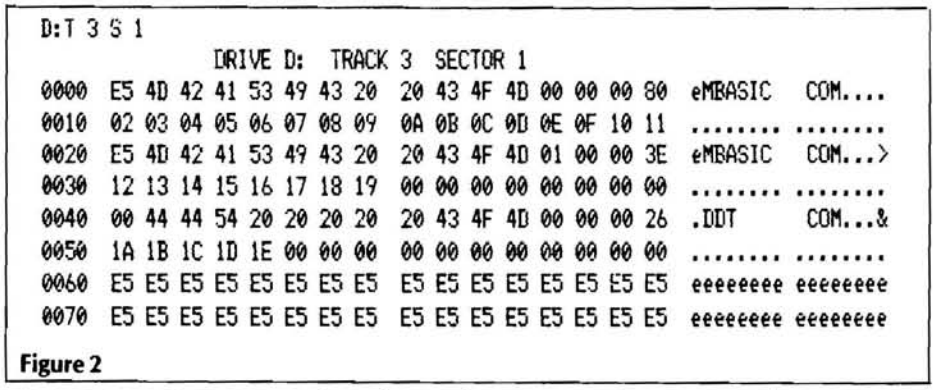
Start looking for your deleted file at the first sector of DIRECT.SYS and continue looking in succeeding sectors until you find it. Figure 1 is a UDUMP display of the directory of a disk containing a deleted file. The output from DUMP is similar. Notice that the left part of the display contains the hex values of the bytes on the sector, and the right part contains the ASCII (textual) interpretation of the data. This disk originally contained MBASIC.ABS and BASIC.ABS, but BASIC.ABS was deleted. You can see MBASIC in the display, but the



```
                          Editing DK1:DIRECT.SYS
  Track:    13                                       File Sector:    0
  Sector:    2                                       File Length:   18


          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
          -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  ----------------
  0000:  4D 42 41 53 49 43 00 00 41 42 53 00 00 03 00 00  MBASIC..ABS.....
  0010:  08 2F 02 15 19 15 19 FF 41 53 49 43 00 00 00 41  ./......ASIC...A
  0020:  42 53 00 00 03 00 00 30 50 02 15 19 15 19 FF 00  BS.....0P.......
  0030:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  0040:  00 00 00 00 00 FF 00 00 00 00 00 00 00 00 00 00  ................
  0050:  00 00 00 00 00 00 00 00 00 00 00 00 FF 00 00 00  ................
  0060:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  0070:  00 00 00 FF 00 00 00 00 00 00 00 00 00 00 00 00  ................
  0080:  00 00 00 00 00 00 00 00 00 00 FF 00 00 00 00 00  ................
  0090:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  00A0:  00 FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  00B0:  00 00 00 00 00 00 00 00 FF 00 00 00 00 00 00 00  ................
  00C0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF  ................
  00D0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
  00E0:  00 00 00 00 00 00 FF 00 00 00 00 00 00 00 00 00  ................
  00F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 FF 00 00  ................
```

Figure 1



```
D:T 3 S 1
                DRIVE D:  TRACK 3  SECTOR 1
0000  E5 4D 42 41 53 49 43 20  20 43 4F 4D 00 00 00 80  eMBASIC  COM....
0010  02 03 04 05 06 07 08 09  0A 0B 0C 0D 0E 0F 10 11  ........ ........
0020  E5 4D 42 41 53 49 43 20  20 43 4F 4D 01 00 00 3E  eMBASIC  COM...>
0030  12 13 14 15 16 17 18 19  00 00 00 00 00 00 00 00  ........ ........
0040  00 44 44 54 20 20 20 20  20 43 4F 4D 00 00 00 26  .DDT     COM...&
0050  1A 1B 1C 1D 1E 00 00 00  00 00 00 00 00 00 00 00  ........ ........
0060  E5 E5 E5 E5 E5 E5 E5 E5  E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeee eeeeeeee
0070  E5 E5 E5 E5 E5 E5 E5 E5  E5 E5 E5 E5 E5 E5 E5 E5  eeeeeeee eeeeeeee
```

Figure 2

deleted file appears as ASIC. This is because HDOS replaces the first character of the file name with the number 0FFH when the file is deleted. If you look at the line of numbers to the right of the ASIC, you can see FF about in the middle of the line. It is followed by 41, which is the value of the letter A, then 53, which is the value of S, etc.

2. Patch the file name back to what it was originally. Just use DUMP or UDUMP to change the FF to the value of the original letter. In our example, that would be 42 for the letter B. Remember which track and sector you are on, because later you will have to remove this patch.

3. Copy the deleted file to another disk. You might think that patching the file name is all you have to do, but unfortunately something else gets changed when you delete a file under HDOS. It is a file called GRT.SYS, which contains a map of which groups of sectors on the disk are used by files, and which ones are free. However, once the file name is patched, the deleted file can be copied to another disk. Use ONECOPY if you have only one drive.

4. Using DUMP or UDUMP, patch the first character of the deleted file back to FF. After you have done this, dismount the disk (if it was mounted) and remount it (or use ONECOPY) and copy the saved file back to it. The file is now restored to the disk from which it was deleted.

RECOVERING DELETED CP/M FILES

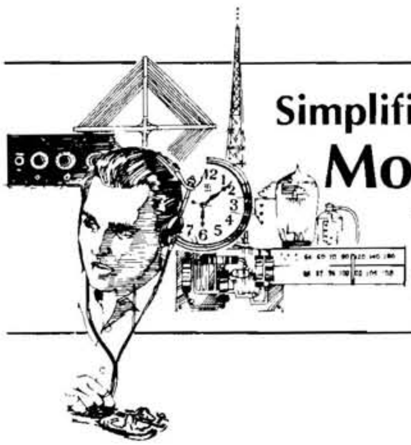
With CP/M, recovering deleted files is a bit easier than with HDOS, because the directory is in a fixed place on the disk, and because the map of where the file is is not a separate file, but rather is part of the directory entry. So all you have to do is restore the directory entry, and the deleted file is recovered. Here are the steps to follow.

1. Locate the deleted file in the directory. In CP/M, the directory is in a reserved area on the disk which starts at track 3 sector 1 for 5-inch single density disks (hard or soft sector, 49 or 96 tpi), and at track 2 sector 1 on every other kind of floppy disk. You can examine the disk starting at the first directory sector until you find your file, but there is an easier way to find it. In CP/M, disks are divided into sections called groups, and the directory always starts with group zero, regardless of what track and sector it occupies. So using SDUMP, examine group zero on the disk. If you do not see your file, try group one.

Figure 2 is the SDUMP display of a disk that originally contained two files, MBASIC.COM and DDT.COM. MBASIC was deleted. The first thing that you will probably notice is that there are two MBASIC entries. This has nothing to do with the file being deleted, but is because of the size of the file. CP/M divides files into 16k segments called extents, and usually there is one directory entry for each extent, so a file bigger than 16k but less than or equal to 32k would have two directory entries. These entries will not necessarily be adjacent as in our example, but may be scattered throughout the directory. On some disk formats, there are two extents per directory entry, so MBASIC would have only one entry in that case.

2. Restore the deleted file's directory entry. After you have found the directory entry (or entries) for your deleted file, all you need to recover the file is to patch the entry to what it was originally. In Figure 2, you will notice a small e before the word MBASIC in each entry. This is SDUMP's ASCII interpretation of the number 0E5H. If you look at the line of numbers to the left of MBASIC, you will see that the first number is E5. To restore the entry, change this E5 to the user number occupied by the file originally. In most cases, files are in user number 0, so you would patch the E5 to 0. Be sure to patch each entry for a given file. Then when you exit SDUMP, you can do a DIRectory on the disk and see that the file is restored.

These methods for recovering deleted files are not completely foolproof. For example, if a file has gone through several revisions (for example, a text file that you have edited several times), there may be extraneous information in the HDOS GRT, or too many directory entries with CP/M. In either case, you may wind up with a file that is larger than the original, containing "garbage" along with the good data. But in most cases, the recovery will be successful.



Simplified Interface for the Morse Code Transceiver Program (P/N 885-1106)

Robert R Anderson, K2BJG
69 Page Dr.
Oakland NJ 07436

Updated HUG software introduced in this issue of REMark includes P/N 885-1106 MORSE CODE TRANSCEIVER VER 1.1. This is a modified version of the previously released MORSE-89 program under the same part number.

The modification eliminates the need for special external hardware to generate pulses at the leading and trailing edge of the receive morse code signal, and also eliminates the need to install internal wiring connecting to circuits within the computer. All I/O is now at RS232C levels via the DTE port. All other features remain the same.

For those who now have the MORSE-89 program and may wish to change to this simplified I/O this article includes in detail all software changes to the MORSE-89 "CW.ASM" file required to produce the new MORSE CODE TRANSCEIVER PROGRAM VER 1.1.

Amateur radio station interface equipment is still required to convert station receiver audio CW signals to RS232C receive CW signals, and to interface RS232C CW transmit signals to the station transmitter. Since the requirement for special pulses for receive operation no longer exists, the commercially built units advertised on the amateur market for this purpose can be used without modification. Just be sure to check RS232C pin vs function connections.

For those who prefer to "roll their own" reference is made to my article published in the November 1981 issue of REMark. The receiver audio to TTL converter remains the same. (See fig 1). Fig 2 shows the revised circuitry for signal routing with the necessary interface for RS232C levels.

The CW receive program operates by polling at clock time (every 2 ms) the status of a flag indicating if the receive CW signal is "ON" or "OFF". This flag must be set "ON" by the leading edge, and reset "OFF" by the trailing edge of the receive CW signal. To accomplish this the MORSE-89 program required an external hardware pulse generator producing leading and trailing edge pulses hard wired to the computer interrupt bus at level 5 and 6. The program then set the flag upon interrupt 5 and reset the flag upon interrupt 6. This method works well provided the input pulse width is correct but causes considerable grief when it is not.

Eliminating this problem is another advantage of the revised method used by the new program. MORSE CODE TRANSCEIVER VER 1.1 uses interrupt 5 to set or reset the flag. The receive CW signal is connected to the CTS input of the DTE I/O port. Upon a change of the CW input signal "ON" or "OFF" the 8250 initiates a level 5 interrupt. The program then checks the CTS input status and accordingly sets or resets the flag.

The CW transmit program in MORSE-89 used the internal general purpose port for output. MORSE CODE TRANSCEIVER VER 1.1 uses the DTR output of the DTE port. A transmit control signal if required is available from the RTS output of the DTE port.

DTE PORT 330Q CONNECTIONS:

PIN 1 — PROTECTIVE GROUND

PIN 4 (RTS) — TRANSMITTER CONTROL OUTPUT (ON DURING XMT)

PIN 5 (CTS) — CW RECEIVE SIGNAL INPUT

PIN 7 — SIGNAL GROUND

PIN 20 (DTR) — CW TRANSMIT SIGNAL OUTPUT

ALL SIGNALS — NEGATIVE LEVEL = OFF, POSITIVE LEVEL = ON

The serial I/O board jumper associated with PORT 330Q must be in position 5. This enables INTERRUPT 5 used by the RECEIVE PROGRAM.

Software Changes:

Edit the CW.ASM file following steps 1 thru 14 below. Delete the file HDOSN.ACM. From the HDOS VER 2.0 disk "SOFTWARE TOOLS" transfer to your disk the files HOSEQU.ACM and HOSDEF.ACM. From the HDOS VER 2.0 disk "DEVICE DRIVERS" transfer to your disk the file U8250.ACM. Reassemble CW.ASM.

Changes to "MORSE-89" CW.ASM File:

Note: *References to line numbers refer to the original lines in MORSE-89 prior to changes. A line insert or line deletion does not, for the purposes of these instructions, change subsequent line numbers.*

```
1. Change the TITLE to read:
           TITLE    "MORSE CODE TRANSCEIVER VER 1.1"

2. Revise "EQU" listing at start of program:
delete:             SC.ACE  EQU    000350A
delete:             UR.LSR  EQU    5

insert:             PORT    EQU    330Q      I/O PORT

3. Revise "XTEXT" listing next in program:
delete:             XTEXT   HDOSN

insert:             XTEXT   HOSEQU
insert:             XTEXT   HOSDEF
insert:             XTEXT   U8250

4. Revise "ENTER" routine:
change line 9 & 10 to read:
                    MVI     A,UC.EDA
                    OUT     SC.ACE+UR.IER
```

```
5. Revise "CCHIT" routine:
insert after line 1:
                EQU     *
                XRA     A
                OUT     PORT+UR.IER
                OUT     PORT+UR.MCR

delete line 11:
                STA     UIVEC+15
delete line 14:
                STA     UIVEC+16

6. Revise "GOTHDOS" routine:
delete line 5:
                STA     UIVEC+15D
delete line 10:
                LXI     H,INT6
delete line 11:
                SHLD    UIVEC+16D

insert after line 13:
                STA     MFLAG
                MVI     A,UC.MSI
                OUT     PORT+UR.IER

7. Revise "INT5" routine:
insert after line 1:
                PUSH    PSW
                IN      PORT+UR.MSR
                ANI     UC.CTS
                JZ      KEYOFF

8. Revise "INT6" routine:
delete line 1:
      INT6    PUSH      PSW

add LABEL to line 2:
                XRA     A
      KEYOFF  XRA     A

9. * Revise "ON" routine:
change line 1 to read:
                MVI     D,UC.DTR+UC.RTS

insert after line 1:
                MVI     E,UC.RTS

10. * Revise "OFF" routine:
change lines 1 & 2 to read:
                MVI     D,UC.RTS
                MVI     E,UC.RTS

11. * Revise "SND1" routine:
change lines 2 & 12 TO READ:
                OUT     PORT+UR.MCR

* delete all comments concerning
  changes for H-8
```

```
12. Revise "MODSW" routine:
insert after line 15:
                DB      040Q,+200Q
                MVI     A,8D
                STA     CHRCNT

13. Revise "CPTD" routine:
change line 19 to read:
        DB      033Q,131Q,054Q,040Q      TO 13-01

insert after line 19:
                DB      012Q

insert after line 22:
                DB      040Q+200Q
                MVI     A,6D
                STA     CHRCNT

14. Revise "RCVLOP" routine:
insert after line 1:
                EI
                XRA     A
                OUT     PORT+UR.MCR
```

Note: - *Steps 1 thru 11, and 14 concern the I/O modification. Steps 12 and 13 correct a "bug" in the MORSE-89 program that caused the length of the first line to be incorrect in both RX and TX modes.*

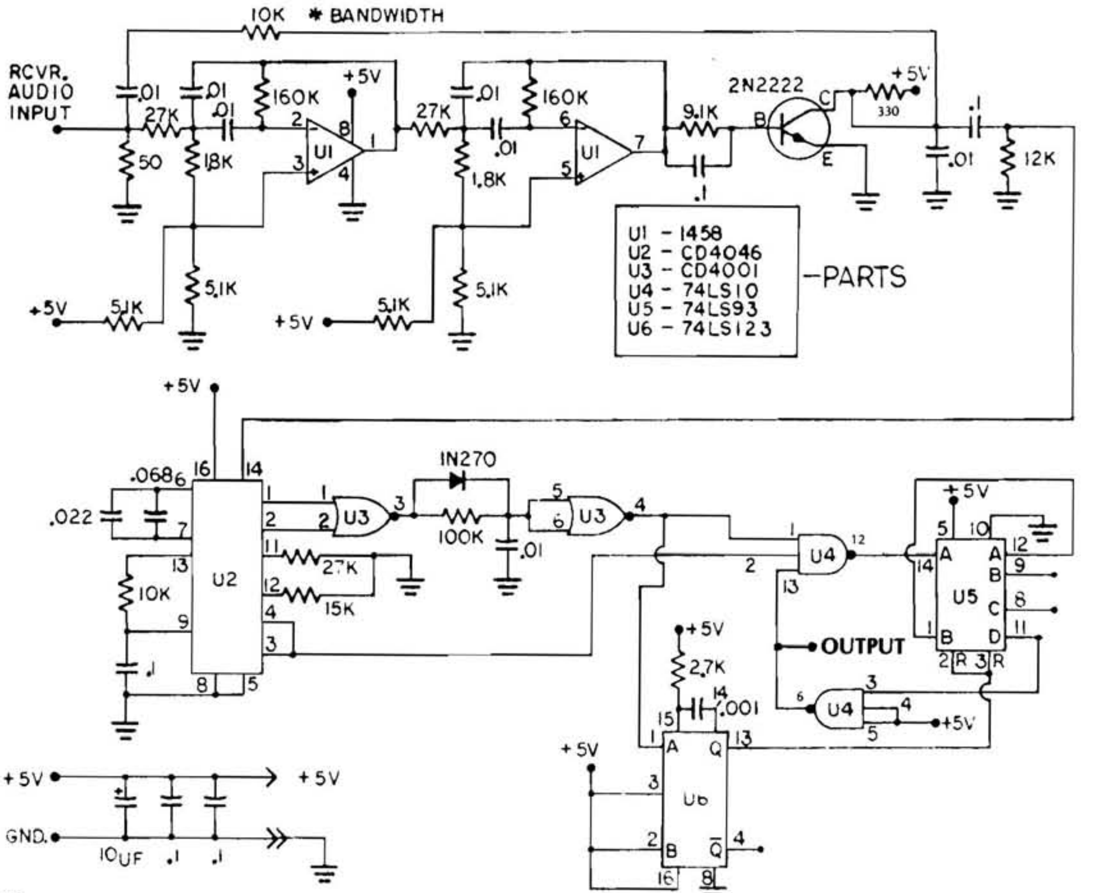


Figure 1

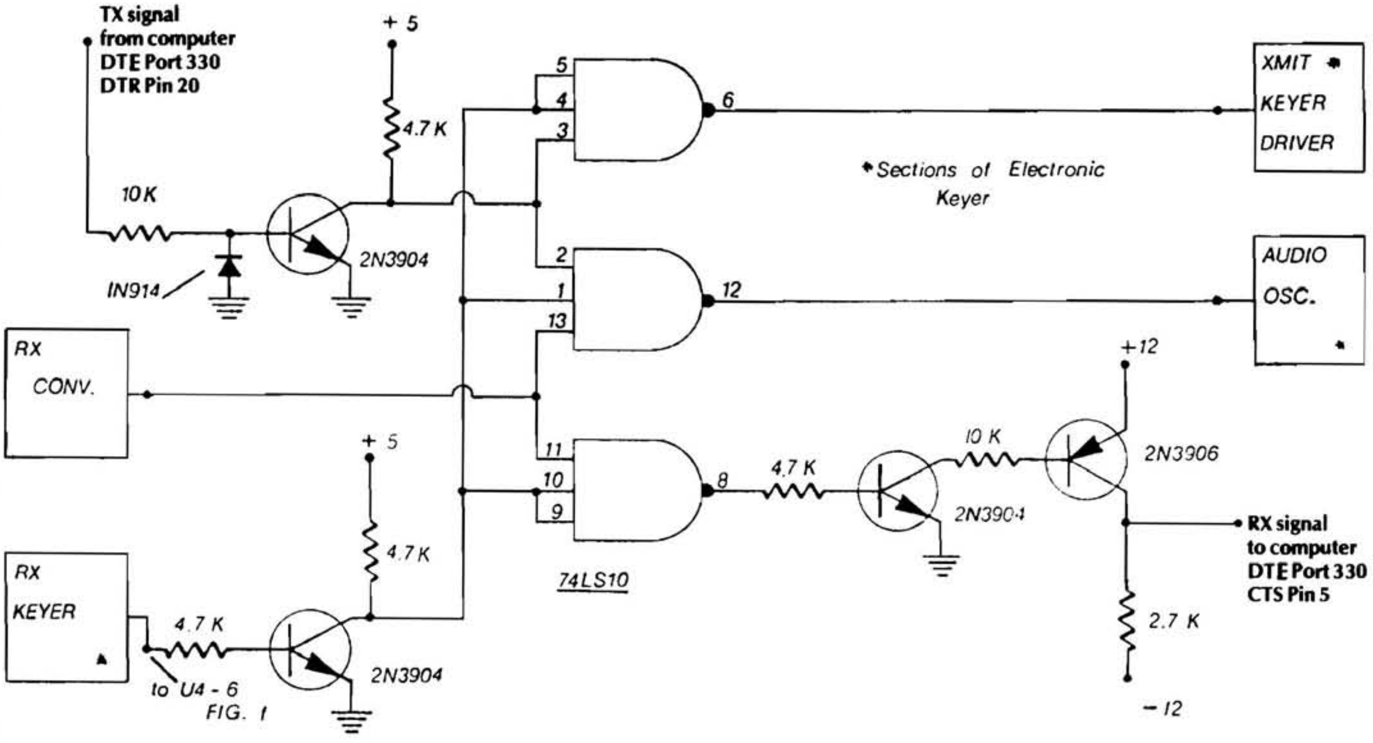


Figure 2

NEW HUG PRODUCTS

885-1120
HDOS "Whew" Utilities $20.00

Introduction: This utilities disk contains a number of programs which will aid the user when he has accidentally erased a file, or run out of disk space, or needs a printout of some important information on the screen, and more.

Requirements: These utilities require the HDOS operating system version 2.0 on an H19/H8/H17 or H89 with 32K of memory. One drive is required for most of the programs (two drives are required for SAVE.ABS). A printer is required for HC.ASM. A daisywheel printer is required for PWHEEL.ABS.

These programs are written in assembly language and the source code is included for most of the programs.

NOTE: *This disk requires the H19 (H89) terminal for some of the programs. Other special requirements are listed above.*

Authors:

HC.ASM — Original program by Walter A. Rison. This version by Burton Hulland of Hulland Engineering,

RAM.ASM — Herbert Drake, Jr.,

UNDELETE, SAVE, SFLAGS, REDUCE, REDUCDIR, OLDSTAT, VIEW, PWHEEL, MON — Patrick Swayne.

UNDELETE — This program can recover a deleted disk file provided that no write operations have been performed on the disk since the file was deleted. In normal use it is the opposite of the HDOS DELETE command.

UNDELETE reads the file into memory and writes it back to the disk as a normal file. If the file is too large to fit into memory, UNDELETE will leave a message and abort the operation. In this case, the user can specify a drive that UNDELETE will use to store the file temporarily while it restores it.

Note: *If deleting two or more files on a disk, only one of them can be safely recovered with UNDELETE, since when it writes the recovered file back it may write over the other deleted file(s). If the deleted file has gone through several re-writes (for example, it has been edited several times), the recovered file may include additional "garbage" sectors. If it is a text file, they can be removed with an editor.*

SAVE — This is a more sophisticated deleted file recovery program than UNDELETE. It saves deleted files by copying them to another drive and can recover several deleted files from one disk. For example, if the user accidentally types DELETE *.BAS, all of the deleted .BAS files can be recovered with SAVE.

When SAVE signs on, it will ask for the drive containing the deleted files, and a drive to save them on (two different drives). Then the user can enter the name of a file to recover, or DIR to see a list of deleted files.

If the user enters DIR, the deleted files on the disk will be listed with a question mark in place of the first character in the file name. This is because HDOS overwrites the first character when it deletes the file. For example, if the deleted files are FILE1.BAS, FILE2.BAS, and FILE3.BAS, they will be listed as:

```
?ILE1.BAS
?ILE2.BAS
?ILE3.BAS
```

To recover a file, the original name is typed in, not a question mark for the first character. (Actually, the question mark can be replaced with any alpha character.)

Note: *There must be at least two drives to use SAVE. The user is restricted to using UNDELETE (with files that will fit in memory) if he has only one drive.*

SFLAGS — This program is a replacement for the FLAGS program supplied with HDOS. Its operation is identical to FLAGS except that it can remove the L (Lock) flag from a file, as well as set it. It can also display the "C" (Contiguous) flag, but it cannot set or clear it.

CFLAGS — This is a special version of FLAGS that can set (but not clear) the "C" flag. It should be used only in conjunction with the REDUCE.ABS program (see next abstract).

REDUCE — This program will make more user space available on HDOS 2.0 system disks. It removes the Patch History Table (PHT) from HDOS system files. (All programs and device drivers provided with HDOS are system files).

The *PHT* is an extra sector appended to the end of the HDOS files, which does not affect their operation. If a file with a PHT uses an odd number of sectors, removing the PHT will free up two disk sectors on a 5-inch single sided, single density disk (hard or soft sector).

Note: *REDUCE.ABS will probably be of little value for disks with a higher capacity than 5-inch single sided single density. It is always advised to have back up copies of anything that is REDUCEd, in case something goes wrong.*

REDUCDIR — This program reduces the size of the file DIRECT.SYS on a disk to the smallest practical size, freeing up several sectors. If it is used on a disk that already contains several files, it will reduce the size of DIRECT.SYS only as much as the unused directory space will allow.

HC — This is the source code for a device driver which works with the regular printer device driver to add two new features to HDOS.

The first feature is a screen dump. When a Control- ↑ (Control-Shift-6) is typed, whatever appears on the H19/H89 screen at the time is sent to the printer.

The second feature, a printer toggle, is turned on and off by Control-\. While it is on, whatever is typed on the screen goes to the printer (like Control-P in CP/M). More complete instructions are contained in the file HC.DOC.

For the users convenience, four ready-to-use assembled versions of HC are included on this disk.

Note: *HC.ASM is a public domain program that may be copied for any reason as long as the author's name is included.*

RAM89 and RAM90 — These are RAM test programs for the H89/Z89/Z90 family of computers. RAM90.ABS is for Z90 computers and any others equipped with MTR90. RAM89 is for all other monitors.

Note: *The test for which monitor the system has is to reset the computer to get the "H:" prompt, and type the letter V. If the word "View" appears, the system has MTR90.*

When running RAM89 or RAM90, a menu will be displayed from which the user can select a 16k area of RAM to be tested. The test is very slow and should be run for at least an hour. If a bad "cell" is found, its address and contents will be displayed.

OLDSTAT — This program provides the STAT command as it exists in HDOS 1.6 for HDOS 2.0 users. It lists all soft (recoverable) and hard (non recoverable) disk errors on the hard sector drives since the last boot. The sector of the last hard error is also listed.

VIEW — This program is a substitute for the TYPE command in HDOS. It is useful in cases where the user has a very large file and wishes to see only a few lines of it.

The TYPE command loads the entire file into memory (or as much as will fit) before starting the screen display, but VIEW loads the file in one sector at a time, which will start the display much sooner.

PWHEEL — For daisywheel printers, this program will make a copy of exactly what is on each of the printwheels. It prints a list of all the characters on the printwheel in two columns so that the list will fit on one page.

The ASCII value of each character is printed beside the character, and the two "hidden" characters on Diablo 96 character printwheels are also printed. PWHEEL also allows the user to enter an argument in the command line to be used for a heading in the printed list.

MON — This program aborts HDOS and returns control to the computer's monitor. On an H89, it returns to the "H:" prompt without clearing the screen.

ROMSUBS.ACM — This file is necessary to assemble some of the source files on this disk. The other .ACM files required are supplied with HDOS 2.0.

Comments: This utilities disk provides excellent programs for helping the user in a number of ways.

885-1221
The WATZMAN/HUG H19 .... $30.00

Introduction: The WATZMAN/HUG H19 is replacement software for the code and keyboard encoder ROMs in the H19, H19A, or Z19 terminal, or the Terminal Logic board in an H89, H89A, Z89, or Z90. It can greatly add to the capabilities and usefulness of the terminal.

The WATZMAN/HUG H19 is available in assembly source code and assembled hex files.

Requirements: The source code programs require the CP/M operating system and are presented on two disks.

885-1221 includes two assembled hex files: MYH19.HEX file for the code ROM, and KEYBD.HEX for the keyboard ROM. MYH19 requires 4k of space and should be programmed into two 2516/2716 five volt only EPROMS or one 2532 EPROM. KEYBD should be programmed into one 2516/2716.

The documentation file describes how to install the new ROMs in the H19 or H89.

Features: The WATZMAN/HUG H19 adds several new features to the H19 terminal. These include:

1) *Greater baud rate selection.* Rates from 75 to 38400 may be selected by software or switch settings, including 134.5.

2) *On screen digital clock.* A digital time display is maintained on the 25th line which does not interfere with terminal operation or use of the 25th line. The clock is controlled by escape codes and can be set and read. The display can be turned on or off.

3) *Native mode keyboard operation.* With this mode, each key produces a unique 8-bit code instead of the 2-code escape sequences produced by some keys.

4) *Added transmit features.* In addition to the ability to transmit a page and transmit the 25th line, the WATZMAN/HUG H19 can transmit the current (cursor) line and transmit the character at the cursor. The single character transmit feature sends the character as an 8-bit code that allows the user to determine whether it is a graphic and/or reverse video character.

5) *Cursor operation.* Software selectable blinking or steady cursor, in underline or block format.

6) *Duplex selection.* Software selectable half or full duplex operation. This makes the H19 more useful in certain remote communication applications.

7) *Software selectable handshaking.* The user can select either the normal Xon/Xoff handshaking or hardware handshaking via RTS (pin 4).

8) *Word length.* Switch selectable 7 or 8 bit word length.

9) *Transparent mode.* In this mode, each 8-bit code from 0 to 255 produces a unique visible character.

10) *Upward compatability.* The escape codes for all of the new features (except 134.5 baud select, handshaking select, and transparent mode) are in accordance with Heath/Zenith standards, so most of the features are compatible with similar features on newer Heath/Zenith products, including the ZT-1 and Z100 series computers.

Comments: The WATZMAN/HUG H19 ROM will greatly increase the usefulness of the H19 terminal line for those who are intrigued with the new features.

HUG PRODUCT UPDATES

The following HUG products have been updated, fixed, and/or modified.

885-1086
HDOS Tiny PASCAL version 4.1 ..... $20.00

This Tiny Pascal is a greatly enhanced version of the original Tiny PASCAL. It includes the new features of LOAD/STORE, CHAIN, random file I/O, octal values from 0 to 377377, better EOF checking, reading/writing of files containing bytes of any value 0 to 255 decimal, lower case and underscore for identifiers, EXIT, and CALL for assembly language interface.

Tiny Pascal System Version 4.1 is a Pascal compiler using only the integer subset and includes the compiler, translator, and generous documentation. Tiny Pascal requires HDOS 2.0 on an H8/H17 or H89 with 32K of memory.

The disk includes a program, CONFIGUR, which will set the defaults of the compiler, translator, and runtime routines. The source code of CONFIGUR is a good example of how to use TPascal random I/O.

Also on the disk are other example programs to aid in understanding

Tiny Pascal. The H19 terminal is required for one of the example programs.

885-1106
MORSE CODE TRANSCEIVER version 1.1
.... $20.00

This program is a modified version of MORSE-89, previously released as the same part number. External interface equipment and I/O connections have been simplified. All I/O is now handled at the DTE port. (Refer to "Simplified Interface for the MORSE CODE TRANSCEIVER Program" by Robert Anderson, page 17 of this issue.)

The disk requires HDOS 2.0 on an H19/H8/H17 or H89 with 32K of memory. Only one disk drive is required. The H19 (H89) terminal is required. Interrupt 5 must be set on the serial I/O board.

Note: *This version eliminates the need for special external hardware to generate pulses at the leading and trailing edge of the receive morse code signal. It also eliminates the need to install internal wiring to circuits within the computer.*

MORSE CODE TRANSCEIVER is a fully documented 8080 assembly language program for the H-89 computer which provides the operator with the ability to send and receive morse code. This program is intended to be used by radio amateurs to facilitate communication by morse code over a wide range of code speeds, dot/dash ratios, signal strengths, and interference noise conditions.

Comments: The article by Bob Anderson includes, in detail, all software changes to update the MORSE-89 program, for those who have purchased same. The latest version will include on the disk label "version 1.1".

For the individual interested in CW operation, MORSE CODE TRANSCEIVER is the package for interfacing the H89 to CW operations.

885-1107
Transaction Data Management (TMS) . $30.00

This is simply an announcement that TMS has been updated to fix the bugs that have been discovered since its release. The sort routine is now fixed and is more efficient than originally projected.

Any user who has purchased P/N 885-1107, may return the original disk, and we will replace it with the updated TMS files and the changes in the documentation. (Do not send the documentation, we will send the changes to the documentation. There are only a few.)

Note: *Returns of this type are normally not done for new versions, unless the updates are to fix existing bugs rather than present new features to the programs, and the patches are too major to be published in REMark.*

885-1110
HDOS AUTOFILE (Z80 code only) $30.00

This is a modification to the AUTOFILE Data Base Management System (DBMS) for organizing, filing, and retrieving information on a suitable scale for personal use.

AUTOFILE version 2.0 has three new features which the author describes on page 9 of this issue. He also includes patches to the old version.

AUTOFILE requires HDOS 2.0 on an H89 with 48K of memory. It makes extensive use of the Z80 instruction set, and therefore will run on H89's or H8 Z-80 based systems only.

Note: *The programs are written using the* Zilog *mnemonics. The source files and .ACM files are included.* UVMAC *release 2.2 from* The Software Toolworks *was used to assemble the modules.*

Comments: Version 2.0 of AUTOFILE is now released on two disks, with one disk containing the source code and XTEXT files required to assemble the programs.

CHARGE......IT!

As many of you already are aware, Heath Company agreed long ago to handle the Heath Users' Group software products as we are simply too small to maintain the appropriate HUG inventory levels within our limited office space. Recently, Heath implemented a MasterCard program allowing customers to purchase Heath parts using Visa or their MasterCard. So, why not HUG?

You guessed it! Many of our users have requested the ability to order HUG products by phone using their Visa or MasterCard. With a few minor changes, HUG has now been included in the Heath program to add this increased convenience for our members. You may order any HUG product or products by phoning the Heath Company Parts Department at (616) 982-3571 ($10.00 minimum order, please). Be sure to have the HUG part number (885-) and product description available along with your HUG ID to ensure prompt delivery of the product you desire.

We wish to thank the Heath Parts Department staff for including the Heath Users' Group in their improved program for parts order entry.

HUG PRODUCTS LIST

NOTE: The number in the REM # column refers to the issue of REMark containing a description of the software. Usually, it refers to the "New HUG Sofware" column, but it may refer to an article.

Part numbers shown in **bold print** are available in soft sector 5.25-inch format. Add -37 to the part number to order soft sector. For example, to order 885-1206 in soft sector, use 885-1206-37.

Part Number	DESCRIPTION	Selling Price	REM #
CASSETTE SOFTWARE (H8 and H88)			
885-1008	Volume I Documentation and Program Listings (some for H11)	$ 9.00	
885-1009	Tape I Cassette	$ 7.00	
885-1013	Volume II Documentation and Program Listings	$12.00	
885-1014	Tape II ASM Cassette H8 Only	$ 9.00	
885-1015	Volume III Documentation and Program Listings	$12.00	
885-1026	Tape III Cassette	$ 9.00	
885-1036	Tape IV Cassette	$ 9.00	8
885-1037	Volume IV Documentation and Program Listings	$12.00	8
885-1039	WISE on Cassette H8 Only	$ 9.00	
885-1058	Volume V Documentation and Program Listings	$12.00	
HDOS SOFTWARE H8/H17 or H89 — (5-inch only)			
MISCELLANEOUS COLLECTIONS			
885-1032	Disk V H8/H89	$18.00	8
885-1044	Disk VI H8/H89	$18.00	
885-1064	Disk IX H8/H89	$18.00	
885-1066	Disk X H8/H89	$18.00	10
885-1069	Disk XIII Misc H8/H89	$18.00	
GAMES			
885-1010	Adventure Disk H8/H89	$10.00	4
885-1029	Disk II Games 1 H8/H89	$18.00	8
885-1030	Disk III Games 2 H8/H89	$18.00	8
885-1031	Music 8 & 89 H8/H19 and H89	$20.00	25
885-1067	Disk XI Graphic Games .ABS and B H BASIC (H19/H89)	$18.00	12
885-1068	Graphic Games (H19/H89)	* $18.00	10
885-1088	Graphic Games (H19/H89)	* $20.00	14

885-1093	Dungeons and Dragons Game Requires H89 or H8/H19	*	$20.00	16
885-1096	Action Games (H19/H89)	*	$20.00	18
885-1103	Sea Battle Game (H19/H89)		$20.00	20
885-1111	HDOS MBASIC Graphic Games	*	$20.00	23
885-1112	HDOS Graphic Games		$20.00	23
885-1113	HDOS Fast Action Games		$20.00	23
885-1114	Color Raiders and Goop (HA-8-3)		$20.00	23
885-8009	HDOS & CP/M Galactic Warrior	*	$20.00	32

UTILITIES

885-1019	Device Drivers (HDOS 1.6)		$10.00	6
885-1022	HUG Editor (ED) Disk H8/H89		$15.00	20
885-1025	Runoff Disk H8/H89		$35.00	
885-1060	Disk VII H8/H89 SUBMIT, CLIST, FDUMP, ABSDUMP, etc.		$18.00	
885-1061	TMI Cassette to Disk H8 only		$18.00	
885-1062	Disk VIII H8/H89 (2 disks) MEMTEST, DUP, DUMP, DSM		$25.00	
885-1063	Floating Point Disk H8/H89		$18.00	
885-1065	Fixed Point Package H8/H89		$18.00	10
885-1075	HDOS Support Package H8/H89		$60.00	
885-1077	TXTCON/BASCON H8/H89		$18.00	
885-1079	HDOS Page Editor		$25.00	15
885-1080	EDITX H8/H19/H89		$20.00	
885-1082	Programs for Printers H8/H89		$20.00	
885-1083	Disk XVI RECOVER, etc.		$20.00	11
885-1089	MACRO, CTOH, and misc Utilities		$20.00	20
885-1090	Misc. HDOS Utilities CCAT, HPLINK, AH, MBSORT, etc.		$20.00	22
885-1092	RDT Debugging Tool H8/H89		$30.00	14
885-1095	HUG SY: Device Driver HDOS 2.0		$30.00	18
885-1098	H8/HA-8-3 Color .ABS/.ASM		$20.00	19
885-1099	H8/HA-8-3 Color in Tiny Pascal		$20.00	19
885-1105	HDOS 2.0 Device Drivers MX-80, Paper Tiger, Clock, etc.		$20.00	24
885-1116	HDOS Z80 Debugging Tool		$20.00	27
885-1119	B H BASIC Support H8/H19 or H89		$20.00	29
885-1120	HDOS "WHEW" Utilities		$20.00	33
885-8001	SE UCSD-Style Screen Editor		$25.00	28
885-8003	B H BASIC to MBASIC Converter		$25.00	28
885-8004	UDUMP and FAKEMNT Disk Manipulation Utilities		$35.00	28
885-8005	MAPLE Modem Program		$35.00	29
885-8006	HDOS SUBMIT		$20.00	31
885-8007	EZI-TRANSfer		$30.00	30

PROGRAMMING LANGUAGES

885-1038	WISE on Disk H8/H89		$18.00	
885-1042	PILOT H8/H89		$19.00	
885-1059	FOCAL-8 H8/H89		$25.00	13
885-1078	HDOS Z80 Assembler		$25.00	21
885-1085	PILOT Documentation		$9.00	
885-1086	Tiny Pascal H8/H89		$20.00	13
885-1094	HUG Fig-Forth H8/H89 2 Disks		$40.00	18

BUSINESS, FINANCE AND EDUCATION

885-1047	Stocks H8/H89		$18.00	
885-1048	Personal Account H8/H89		$18.00	
885-1049	Income Tax Records H8/H89		$18.00	
885-1055	Inventory H8/H89	*	$30.00	
885-1056	Mail List H8/H89	*	$30.00	
885-1070	Disk XIV Home Finance H8/H89		$18.00	
885-1071	Small Business Package III 3 Disks H8/H19 or H89	*	$75.00	17
885-1091	Grade and Score Keeping	*	$30.00	14
885-1097	Educational Quiz Disk H89 or H8/H19	*	$20.00	18
885-1118	Payroll	*	$60.00	30
885-8010	HDOS Checkoff		$25.00	32

DATA BASE MANAGEMENT SYSTEMS (DBMS)

885-1107	Amateur Radio Logbook and TMS		$30.00	23
885-1108	Telephone/Mail Info. System	*	$30.00	23
885-1109	Retriever (2 disks)		$40.00	23
885-1110	Autofile		$30.00	23
885-1115	Aircraft Navigation H8/H89	*	$20.00	25
885-8008	Farm Accounting System	*	$45.00	30

AMATEUR RADIO

885-1106	Morse-89 H8/H19 or H89		$20.00	22

*Means MBASIC is required

H11 SOFTWARE

885-1008	Volume I Documentation and Program Listings (some for H11)		$ 9.00	
885-1033	HT-11 Disk I		$19.00	
885-1053	H11/H19 Support Package EXEC Modem Software, etc.		$20.00	27
885-1117	Pirate's Adventure for H11/H19		$20.00	28

CP/M SOFTWARE (5-inch only)

885-1201	CP/M (TM) Volumes H1 and H2	%	$21.00	
885-1202	CP/M Volumes 4 and 21-C	%%	$21.00	
885-1203	CP/M Volumes 21-A and B	%%	$21.00	
885-1204	CP/M Volumes 26/27-A and B	%%	$21.00	
885-1205	CP/M Volumes 26/27-C and D	%%	$21.00	

The above CP/M products are 2 disks each.

885-1206	CP/M Games Disk	*	$20.00	11
885-1207	TERM and H8COPY		$20.00	26
885-1208	HUG Fig-Forth H8/H89 2 Disks		$40.00	18
885-1209	Dungeons and Dragons Game MBASIC and H89 or H8/H19	*	$20.00	19
885-1210	HUG Editor		$20.00	20
885-1211	Sea Battle Game for CP/M		$20.00	20
885-1212	CP/M Utilities I		$20.00	21
885-1213	CP/M Disk Utilities		$20.00	22
885-1214	Amateur Radio Logbook	*	$30.00	23
885-1215	BASIC-E		$20.00	26
885-1217	HUG Disk Duplication Utilities		$20.00	26
885-1218	CP/M MBASIC Payroll	*	$60.00	31
885-1219	CP/M Aircraft Navigation H8/H89	*	$20.00	31
885-1220	CP/M Action Games	*	$20.00	32
885-1221	Watzman ROM Source Code (2 disks)		$30.00	33
885-8011	CP/M Checkoff	*	$25.00	32

% Means CP/M 1.43 only (ORG-4200).
%% Means CP/M 1.43 or 2.2 (Heath).
MBASIC programs on these disks are for version 4.8 or earlier.
Other CP/M disks are for 2.2.
* means MBASIC is required.

MISCELLANEOUS

885-4	HUG Binder		$ 5.75	
885-4001	REMark VOLUME I		$20.00	23
885-4002	REMark VOLUME II		$20.00	

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

CP/M is a registered trademark of Digital Research Corp.

NEXT MONTH

The Next Step

"C"

Plus many more

INTERESTING ARTICLES

ADVENTURE MAKER

Create your own Adventure type games! Even has provisions for voice output when used with text-to-speech synthesizers! ADVENTURE MAKER allows you to design short or long games, with varying levels of difficulty and almost unlimited possibilities. Several premade Adventure games included. Two separate modules are provided; the Create program and a Run-Time module. Typical Adventure type commands are recognized such as 'pick up bottle', 'use gun', 'drop torch' as well as direction commands. Requires HDOS 2.0, 64K, MBASIC, Z89 or Z19. $21.

BIOS-80 Allows use of 96tpi and 48tpi drives using hard-sector controllers. CP/M only - $35

MONEY$WORTH Logs income & expense transactions for home or small business. CP/M or HDOS - $35

PROBE Random access database system with 12 fields and input form mask. CP/M or HDOS - $29

QUIK-REF Fast MBASIC variable, function and statement line number xref. CP/M or HDOS - $20

COMPACTA Conserves disk space on high density formats such as 96tpi drives. HDOS only - $25

CAVERNS Explore the 30 caverns in this talking (optional) adventure game for Mbasic. HDOS only - $23

UDUMP Versatile Octal, ASCII or HEX screen editor utility for use with any disk format. HDOS - $34

TERMINAL Programmable modem interface with printer trace and Smartmodem features. HDOS - $25

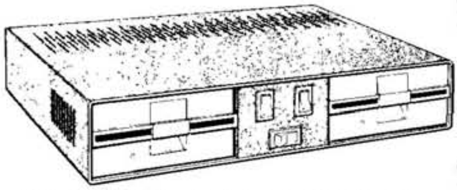
KEY-IT! Keyword in Context database cross xref generator for books, magazines, etc. Runs under Mbasic. HDOS - $25, CP/M - $27

FINANCIAL Amortization, rate of return and cost of borrowing. Runs under Mbasic. HDOS only - $20

MATH WIZ Mbasic elementary math tutor program which checks intermediate steps as well as the final answers. HDOS or CP/M - $19

THE WORD 40,000 word spelling checking program with unique utilities. CP/M only - $69. THE WORD PLUS - $125

SUPERVYZ Menu generator allowing for complete turnkey operation. CP/M only - $89



KS-5000 5¼" dual-drive horizontal enclosure/power supply with front panel power & write protect switches, quiet fan and cables. Designed for 96 tpi drives. $195
KS-TRK1 5¼" single drive vertical enclosure and power supply. $85
KS-TRK2 5¼" dual-drive vertical enclosure and power supply. $120

Z100 Desktop computers from Zenith. Please call for specifications and our low prices.
Z90, Z89 Computers and $1000 of free software. Call for prices.
Z67 Winchester 11 meg hard disk with 8" drive back-up. $4825

TM-100-4 Tandon 96 tpi dual-sided disk drives. $385
TM-100-2 Tandon 48 tpi dual-sided disk drives. $295
TM-100-1 Tandon 48 tpi single-sided disk drives. $225
TM-848-2 Tandon 8" Thinline double-sided disk drives. $515

SMARTMODEM 1200 & 300 baud auto-answer/auto-dial modem. $545
SMARTMODEM 300 baud only, auto-answer/auto-dial modem. $229
CHRONOGRAPH RS-232 clock & calender with timer. $199
D-CAT Direct modem. $165
CAT Accoustic modem. $155

MEMOREX DISKETTES (10/box)
5¼" SS/SD, 10 sector. $26
5¼" DS/DD, 10 sector, hubs. $38
8" SS/SD, soft-sector. $32
8" DS/DD, soft-sector. $44

Z99·G Graphic Printer

This high quality American made dot matrix printer features serial and parallel interface, 9 wire print-head for true descenders, dot-addressable graphics for plotting or creating your own character or graphic sets. 80, 96 and 132 column printing with serif style fonts and expanded print modes. High quality correspondence printouts using an 11 x 9 matrix is software selectable. Bidirectional and 100 cps printing with tractor feed and friction feed are standard! User defineable type fonts and Z89 graphic set in both positive and reverse form using the Style Writer CP/M software. (Shown with the optional QT sound cover and single sheet feeder.)

ZENITH

As a Zenith dealer, we stock many of the original hardware and software products offered by Heath & Zenith, all at reduced prices. A sample of some of the items include Z100, Z89 and Z90 computers, CP/M, Despoolers, Microsoft Basic, 16K expansion boards, DataStar, Supersort, Z25 printers, Pascal, Supercalc, CBASIC, Fortran, Z19 terminals, Wordstar, Magic Wand and more. We also carry some selected non-Zenith software and hardware, such as the Spellbinder word processing system, Spellguard, Supervyz menu generator for CP/M, voice synthesizers and low cost 8" disk controllers for the H8 and H89. We stock Tandon 48 & 96 track per inch, single or dual-sided 5" and 8" disk drives and enclosures - at great prices with full support!

Call, write or use the reader service card to receive our complete 16 page illustrated catalog.

Since 1978 we've built a reputation on quality software and hardware products at low prices, with fast delivery and full support. Our software may be purchased from your local Heathkit Electronic Center or directly from us.

We accept personal or company checks as well as money orders. C.O.D. orders are available for an additional $2 charge. Add $2 shipping and insurance on orders under $100 in the continental United States for UPS delivery. Orders over $100, please add $5 for shipping and insurance. APO and FPO orders add 5% shipping ($2 min.). Foreign orders add 15%. We ship most items within 24 hours upon receipt of your order.

Keyboard Studio inc.
125 Aspen, Birmingham, MI 48009
(313)-645-5365

A 56K Memory Expansion
For The Heath/Zenith 89 Computer

Leon A. Wittwer
7101 Mint Place #204
Alexandria, VA 22306

EDITOR'S NOTE: Please be aware that,with this modification as well as any other hardware modification not obtained directly from Heath Company, the Service Department at Heath will refuse to do ANY service work to correct problems that might arise with your computer. Be sure you read and understand fully this modification before proceeding. ***ATTENTION:*** *If you are uncomfortable using a soldering iron or do not fully understand what is discribed here do not attempt this modification.*

I. Introduction

This article describes a simple, inexpensive upgrade to H/Z89 computers with 64 kilobytes of ram that results in 56 kilobytes of extra ram accessible either directly from within HDOS Microsoft Fortran programs using the MYECS or MYOECS routines or as a fast "disk" using the EC.DVD driver. Both access techniques require the use of the Heath Disk Operating System. The key to the simplicity of this upgrade is that the pin assignments for the 64 kilobit 2164 dynamic memory chip is almost the same as the 16 kilobit chips used in the H/Z89. Another important factor is the software control of the bank 0 memory addresses which greatly simplifies the addressing of the new ram. The following sections provide the upgrade and details for use of the new memory.

II. Hardware Modifications

The H/Z89 computer with 64 kilobytes of ram has the capability to run either the HDOS or CPM. While running HDOS, the lower 8 kilobytes of memory contain the Monitor, the H17 ROM, and 1 kilobyte of static ram for use by the H17 ROM and HDOS. When CPM is run, bit 5 of general output port 362Q is set to 1. This disables the Monitor, the H17 ROM, etc. and enables 8 kilobytes of ram on the 16K MEMORY EXPANSION ACCESSORY, Model WH88-16. This permits 0 origin code for CPM. There is a memory address quirk in this scheme. Writes to the static ram when in the HDOS mode simultaneously writes to the ram on the 16K MEMORY EXPANSION ACCESSORY. The first steps to implement additional memory corrects this addressing quirk by eliminating the double write.

1. Remove the CPU Logic Board from the computer and place component side down.

2. Disconnect pins 12 and 13 of U553 from ground by cutting the connecting lines adjacent to the pins.

3. Cut the line going upwards from pin 4 of RP507. This line ends 1/2 inch above pin 4 and continues on the component side of the board. This line ends on pin 19 of P509, P508, and P507.

4. Connect a wire to the line going upwards from pin 4 of RP507 above the cut made in step 3. Connect the other end to pin 11 of U553.

5. Connect a wire between pin 4 of RP507 and pin 13 of U553.

6. Connect a wire between pin 12 of U553 and pin 6 of U501.

7. Connect a wire between pin 5 of U501 and pin 6 of U517.

8. Insure that the new wires do not stick out from the board and conflict with the Terminal Logic Board.

9. Replace the CPU Logic Board and insure that the computer runs normally. The changes should not affect normal operation.

This completes the fix for the memory address problem.The above wiring disables the RAS for the **memory expansion accessory** whenever U516, the page decoder for bank 0, is selected.

Three wires must be run from the back of the CPU logic board to the **16K memory expansion**. For safety, shielded cables should be used. Two of these wires run from two unused outputs of the general output port 362Q and provide the two additional address lines necessary to address the new memory. The third line controls the multiplexing of the extra address bits into the memory chips.

10. Connect wire A to pin 19 of U514.

11. Connect wire B to pin 6 of U552.

12. Connect wire C to pin 9 of U552.

13. Run all three wires up over the top of the CPU Logic Board and replace the CPU Logic Board.

The remaining steps are on the **16K memory expansion** board, Heath Part No. 181-3372-1. Refer to the instructions for the 16K memory expansion particularly the X-ray diagram on page 7.

14. Mount a 20 pin socket the component side of the board. Mount it below C17 and C16. Put Heath Part No. 443-824(74LS241) into the socket. This part multiplexes the two new address lines into the 8th address pin(pin 9) of the 2164's.

15. Mount a 14 pin socket near the 20 pin socket. Put a 14 pin header plug into the socket. This provides a disconnect for the lines coming from the back of the board.

16. Cut the two lines that pass below U1 and U2 and above C15 and C14 at a point midway between U1 and U2. The upper line is a 5 volt power line. Beyond U1 this line becomes the address line for the 8th address pin of the new memory chips. The second line is the 12 volt power line which no longer is needed on the new memory. It still serves U1.

17. Cut the line going from pin 1 of U1 to pin 1 of U2 immediately adjacent to U1. The line must be cut before it connects to the line on the opposite side of the board that goes to pin 1 of U6. This isolates pin 1 of the new memory that is not used.

18. Cut the line leading from the non-grounded end of C13 before it forks near pin 8 of U9. This removes a capacitor from the new address line.

19. Replace U2 through U9 with 64 kilobit 2164 dynamic memory chips.

20. Connect wire A from the back of the CPU Logic Board to pin 1 of the header plug. If you are using shielded cable, connect the ground to pin 14 of the header plug. This wire multiplexes the new address lines onto the board.

21. Connect wire B from the back of the CPU Logic Board to pin 4 of the header plug. If you are using shielded cable, connect the ground to pin 11 of the header plug. This is the first new address line.

22. Connect wire C from the back of the CPU Logic Board to pin 7 of the header plug. If you are using shielded cable, connect the ground to pin 8 of the header plug. This is the second new address line.

23. Move to the back of the board. Connect pins 8 through 14 of the header plug, pins 4,6,8,10,11,13,and 15 of 443-824, and pin

1 of U2 to ground. This step provides grounds for the shielded cables, grounds unused inputs to 443-824, and grounds pin 1 of the new memory chips.

24. Connect pin 20 of 443-824 to the ungrounded end of C17 to provide 5 volts.

25. Connect pin 1 of the header plug to pins 1 and 19 of 443- 824. This is the driver to the multiplexer(443-824).

26. Connect pin 4 of the header plug to pin 2 of 443-824. This is an address line.

27. Connect pin 7 of the header plug to pin 17 of 443-824. This is an address line.

28. Connect pin 3 to pin 18 of 443-824. This is one of the address outputs.

29. Connect a 33 ohm resistor between pin 18 of 443-824 and pin 9 of U4. The output of pin 3 or 18 provides the address for the 8th address pin(pin 9) of the new memory. The resistor probably provides some noise protection.

30. Connect pin 9 of U1 to pin 8 of U2. This provides the 5 volt power line for the new memory.

31. Cut the line going from the ungrounded end of C19 just before if forks to pin 8 of U1 and the ungrounded end of C14. This completes the disconnecting of the 12 volt power line from the new memory.

32. Cut the line leading from the ungrounded end of C9 before it reaches the "T". This removes another capacitor from the new address line.

33. Cut the line leading from the ungrounded end of C10 before it reaches the "T". This removes another capacitor from the new address line.

34. Cut the line leading from pin 9 of U6 before it connects to anything. This removes another capacitor from the new address line.

35. Cut the line leading from pin 9 of U8 before it connects to anything. This removes the last capacitor from the new address line.

36. Connect pin 1 of U1 to the ungrounded end of C9. This provides a despiking capacitor for pin 1 of U1.

37. Reassemble computer.

This completes the hardware modifications necessary to add the extra ram.

Parts List:

Description	Quantity
1. 2164 64 kilobit dynamic ram chip	8
2. insulated wire	20 in
3. shielded cable	54 in
4. 20 pin socket	1
5. 443-824(74LS241) chip	1
6. 14 pin socket	1
7. 14 pin header plug	1
8. 33 ohm resistor	1

III. Accessing The New Memory

The extra memory is accessed two ways.The memory can be used as a fast "disk" with a data capacity of 218 sectors. This address capability is enabled by using the EC.DVD driver. This driver allows all normal disk manipulations except initialization which is done automatically when the driver is loaded. Initialization takes a few seconds so be patient. If the driver is loaded by a LOAD command, the disk is permanent until the next boot. If a MOUNT command comes first, the driver is loaded but only until a DISMOUNT. After a DISMOUNT, the disk is destroyed. A MOUNT will reinitialize the disk. Thus the entire disk can be reinitialized by a DISMOUNT/MOUNT sequence if the driver was not explicitly loaded.

The second way to use the extra memory is by direct access from within a HDOS Microsoft Fortran program. The software that permits this capabiltiy is titled MYECS. This routine can be assembled and linked to a Fortran program with the Microsoft linking loader. The specifics of using the memory is documented within the source file for MYECS. A second version of MYECS called MYOECS is also available to access the extra memory. This routine differs from MYECS in being compatible with overlaying procedured described in FLOPS, AN ENHANCED CAPABILITIES PACKAGE FOR HDOS MICROSOFT FORTRAN by the author.

Any problems or questions should be referred to:

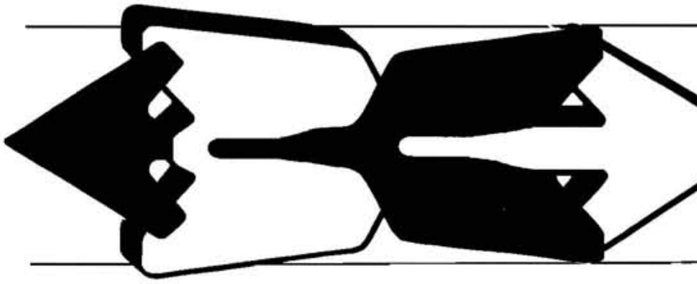
Leon A. Wittwer
7101 Mint Place #204
Alexandria, Va 22306

Telephone: 703-765-9072

This article was taken from a past issue of the >CHUG Newsletter, Capital Heath Users' Group, Inc., P.O. Box 2653, Fairfax, VA 22031

About The Author:

Leon Wittwer is orginally from Monticello, Wisconsin. He graduated from the U.S. Air Force Academy in 1969 and received his doctorate in applied science and engineering in 1972 from the University of California, Davis/Livermore. He is presently a Major with the Air Force, stationed in Washington, D.C. Much of his experience with computers comes from solving a variety of physics problems on mainframe equipment using Fortran, hence the interest in optimizing his H/Z-89 in both hardware and software for number crunching. His system is over a factor of ten faster than standard.



CP/M Console Swapper

Bob Ellerton
HUG Manager

The following program was contributed to the Heath Users' Group as a demonstration of interchanging consoles (terminals) on one CPU. This program can be used with any combination of Heath/Zenith 8080/Z80 based computers (e.g. H8 and two H19s or the H-89 with an additional H19, etc.).

Program Description

The Console Swapper, as written, prints the message "CONSOLE AVAILABLE" on the screen of both terminals connected to the chosen computer. Very simply, the first carriage return from either of the terminals will result in control of the CPU from the terminal that issued the return. A message will then be displayed on the alternate terminal indicating "CONSOLE IS NOT AVAILABLE". To relinguish control, the program is run again from the command terminal (the first to gain control of the CPU). Once again, the message "CONSOLE AVAILABLE" will appear on both terminals completing the cycle. If an operator using the disabled console desires to use the system, a carriage return can be issued, however, control will not be given until the program is run. Therefore, you need not worry about your work being interrupted by the disabled alternate terminal.

Comments

This unique look at the I/O byte under CP/M will allow us to use a little creativity to build some fairly complex uses of the Console Swapper as a subroutine. Can you imagine, for example, the game Battleship with two players viewing alternate screens for the competition? How about using the Console Swapper for the business environment where you may find it necessary to be in two rooms during the normal activity of the day? What about using the Console Swapper to provide a teacher/student communication link? It would appear that this little program could be inserted into many user friendly applications packages as an option or a permanent component of a multi-terminal installation.

Program

```
;
;       CONSOLE SWAPPER PROGRAM
;
;       COPYRIGHT 1982 BY
;       Marc O. Aagenas
;
;
;This program is for use on a Heath/Zenith 8080/Z80
;based computer system using H/Z-19's as both the
;console CRT and the TTY.
;
;
CONIN   EQU     01H
CONOUT  EQU     02H
BDOS    EQU     05H
GIOBYTE EQU     07H
SIOBYTE EQU     08H
PSTRING EQU     09H
CONSTAT EQU     0BH
        ORG     0100H
;
;
START   CALL    M1                      ; PRINT MESSAGE
        CALL    RCON                    ; CLEAR ANY GARBAGE
        CALL    IOSWAP                  ; SWAP I/O BYTE
        CALL    M1                      ; PRINT MESSAGE
        CALL    RCON                    ; SEE IF OTHER USER WANTS SYSTEM
        ORA     A                       ; RIGHT AWAY
        JNZ     SKIP                    ; IF HE DOES THEN GIVE IT TO HIM !!
LOOP    CALL    IOSWAP                  ; SWAP I/O BYTE
        CALL    RCON                    ; READ A CHARACTER
        ANI     0FFH                    ; TEST FOR VALID RESPONSE
        JZ      LOOP                    ; NO THEN SWAP I/O BYTE AND DO IT AGAIN
SKIP    CALL    IOSWAP                  ; YES THEN SWAP I/O BYTE
        CALL    M2                      ; AND WARN OTHER USER
        CALL    IOSWAP                  ; SET UP I/O BYTE TO REQUESTING USER
        RET                             ; RETURN TO SYSTEM
;
M1      MVI     C,PSTRING
        LXI     D,MESS1
        CALL    BDOS
        RET
;
M2      MVI     C,PSTRING
        LXI     D,MESS2
        CALL    BDOS
        RET
```

(ED: To implement this program, you should set up TTY: to the baud rate and port you want in CONFIGUR option A, then assign the PUN: device to TTY: in option C. Connect the second terminal to the port you have selected. Use port 320 (octal) if you have a three port interface. If you only have two ports, use 340 and make sure LST: is not assigned to that port.)

A Data Manager Tutorial Program In B. H. BASIC

D.C. Shoemaker
2000 A Foxridge
Blacksburg, PA 24060

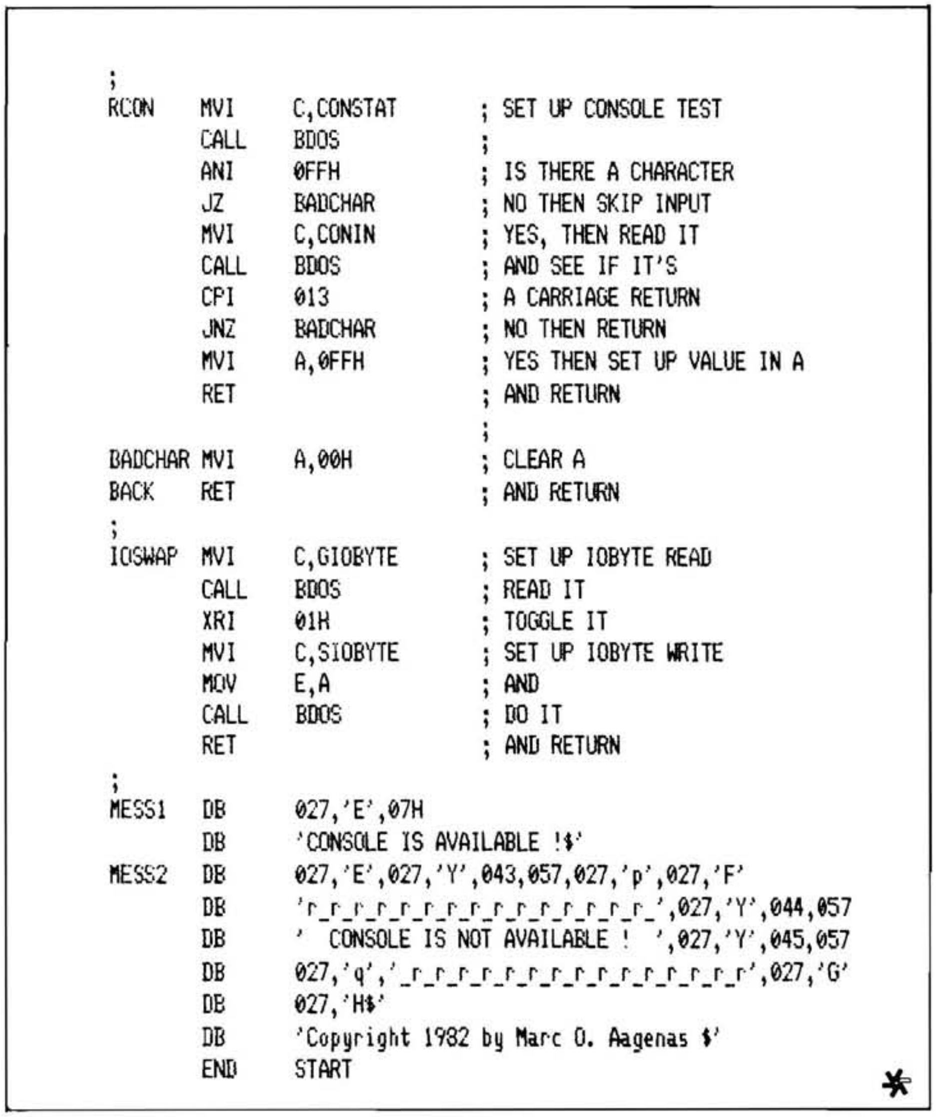
This program is a simple data handler and manager written in Heath's Benton Harbor BASIC for those who don't have access to some of the more powerful file handling capabilities of Microsoft BASIC. It makes use of the unique features of BH BASIC to read data files, determine the end of file (EOF) and write files to the disk. While simple, this program can be elaborated upon to make it do the kinds of data retrieval jobs you need. It's limited only by disk space, and if you have a two-disk system, several hundred files may be maintained.

You'll notice that as now set up, the program is a recipe file handler. This must be the classic example of a program written to justify the new computer to the owner's wife, and the usual resulting effort is rarely if ever used.Let me assure you that this one really works, and is actually used to keep recipe files. The real purpose of the article, however, is to give you a chance to "break in" to data management in a relatively simple and easy manner. Realistically, no program of this sort is of much use until you have hundreds of pieces of information to keep track of. Nevertheless, it can be a good learning tool. Add a routine to sort and merge records, and format output reports, and you're beginning to simulate the professional programs that sell for hundreds and thousands of dollars. Remember, you get what you pay for (usually) and the price of this one is low.

This is an example of the sort of data program called a "key- word" retriever. In essence, it allows the user to enter a set of one-word keys like BREAD, TUNACASS, CHOCCAKE and so on. This key-word file can be read by the program to determine if a given file is present. If so, it can then go out to the disk, locate that file, read it into memory and display it on the console crt. New key-words can be added at any time, and existing files can be updated (replaced) easily.

It probably hasn't escaped your notice that by changing a few words and prompts, this program can just as easily keep track of disk files, books, articles or anything else where a key-word could be used to retrieve a text description. Since there's no real limit to the number of lines of text in the recipe file, this particular program may be more versatile in certain applications than others. For instance, in addition to giving the location of an article reference, the text file could also supply a synopsis of the article, and give further references.

Note that there are two locations within the program where the main directory file (DISKCAT.DAT) is referred to. When you first run the program, if that file isn't present the program will fail. You may either delete lines 10310 and 10480 and create an empty DISKCAT.DAT file, or use an editor to create both DISKCAT.DAT and a data file. Figures



```
;
RCON    MVI     C,CONSTAT       ; SET UP CONSOLE TEST
        CALL    BDOS            ;
        ANI     0FFH            ; IS THERE A CHARACTER
        JZ      BADCHAR         ; NO THEN SKIP INPUT
        MVI     C,CONIN         ; YES, THEN READ IT
        CALL    BDOS            ; AND SEE IF IT'S
        CPI     013             ; A CARRIAGE RETURN
        JNZ     BADCHAR         ; NO THEN RETURN
        MVI     A,0FFH          ; YES THEN SET UP VALUE IN A
        RET                     ; AND RETURN
                                ;
BADCHAR MVI     A,00H           ; CLEAR A
BACK    RET                     ; AND RETURN
;
IOSWAP  MVI     C,GIOBYTE       ; SET UP IOBYTE READ
        CALL    BDOS            ; READ IT
        XRI     01H             ; TOGGLE IT
        MVI     C,SIOBYTE       ; SET UP IOBYTE WRITE
        MOV     E,A             ; AND
        CALL    BDOS            ; DO IT
        RET                     ; AND RETURN
;
MESS1   DB      027,'E',07H
        DB      'CONSOLE IS AVAILABLE !$'
MESS2   DB      027,'E',027,'Y',043,057,027,'p',027,'F'
        DB      'r_r_r_r_r_r_r_r_r_r_r_r_r_r_r_',027,'Y',044,057
        DB      '  CONSOLE IS NOT AVAILABLE !  ',027,'Y',045,057
        DB      027,'q','_r_r_r_r_r_r_r_r_r_r_r_r_r_r_r',027,'G'
        DB      027,'H$'
        DB      'Copyright 1982 by Marc O. Aagenas $'
        END     START
```

*

1 and 2 show two sample data files to illustrate the files' contents. If you wish, you may just create these two files and continue along. Once present, these files can be updated or rewritten with more useful data.

Deleting a file requires using an editor to delete the appropriate entry in DISKCAT.DAT. Then delete that data file from the disk. For my purpose, file deletions aren't a problem, but if they are for you, you might think about rewriting the program to read in DISKCAT.DAT, find the entry to be deleted, delete it and write the file back out to the disk. As they say in the programming manuals, this will be left as an exercise for the reader.

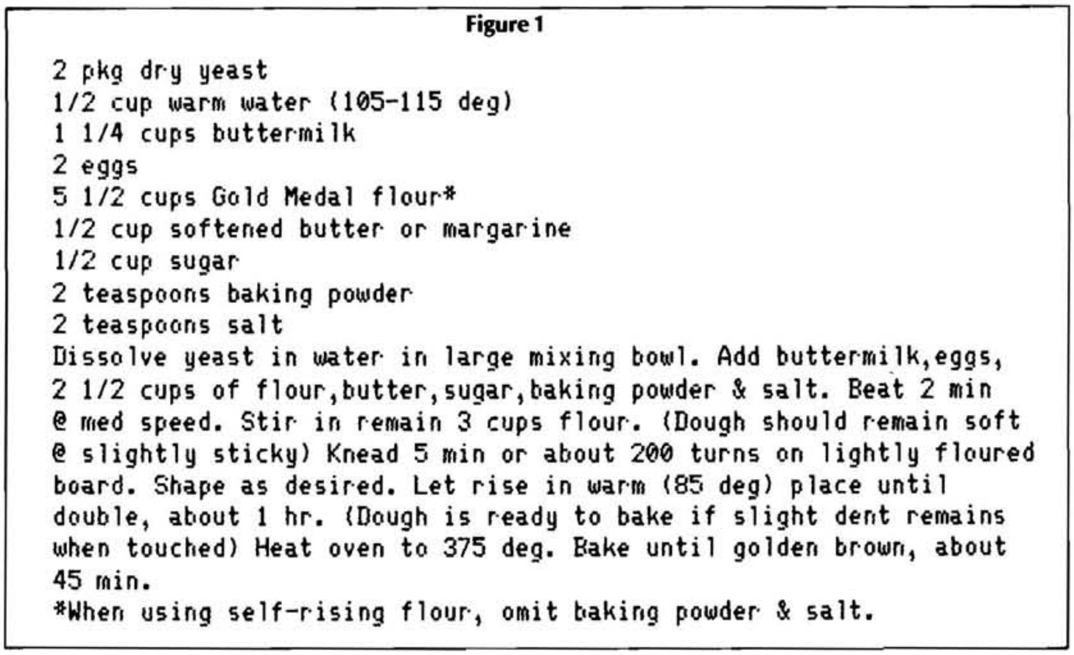
In summary, date management doesn't have to be all that difficult. The techniques used here can start you on the way. Take this program and build on it, changing it to handle the kinds of information you need to store and retrieve.

```
10000 REM This program is a data manager presently configured to work as a
10010 REM recipe file on the H89 computer.  UIt can be easily altered to keep
10020 REM track of any data where you wish to retrieve on one key word.
10030 REM Note that the file DISKCAT.DAT and at least one data file must
10040 REM exist for the program to run.  These may be created by the program
10050 REM itself, or off-line with a text editor.
10060 :
10070 REM Written in B.H. BASIC by D.C. Shoemaker
10080 :
10090 PRINT CHR$(27);CHR$(69):REM Erase screen for H19/H89
10100 PRINT TAB(12);"* * * Recipe File Catalog Program * * *":PRINT
10110 PRINT "    This program saves and retrieves recipes.  You may:":PRINT
10120 PRINT "List                 Print a current list of recipes on file"
10130 PRINT "Retrieve             Retrieve a particular recipe"
10140 PRINT "Update               Enter a new recipe file"
10150 PRINT "End                  End the program"
10160 PRINT :LINE INPUT "Enter your choice, using capital letters:  ";A$
10170 IF LEFT$(A$,1)="E"THEN 10220:REM End the program
10180 IF LEFT$(A$,1)="U"THEN PRINT CHR$(27);CHR$(69):GOTO 10280:REM Update
10190 IF LEFT$(A$,1)="R"THEN 10530:REM Retrieve routine
10200 IF LEFT$(A$,1)="L"THEN PRINT CHR$(27);CHR$(69):GOTO 10730:REM List routine
10210 GOTO 10160:REM Not a valid entry; re-try
10220 END
10230 :
10240 REM Update routine.  When creating the initial file of directory names
10250 REM delete lines 10310 & 10480 to avoid searching for a file that doesn't
10260 REM exist.  Name this master file 'DISKCAT.DAT' & reload 'RECIPE.BAS'
10270 :
10280 PRINT "    To create a recipe file, enter the recipe name as a word of"
10290 PRINT "up to eight letters, then enter the recipe on the lines prompted."
10300 PRINT :LINE INPUT "What is the recipe name?  ";F1$:A$=F1$
10310 GOSUB 11030:REM Delete this line when creating 'DISKCAT.DAT'
10320 IF M1=0 THEN 10360:REM No match found, ok to create new directory
10330 IF M1>0 THEN PRINT CHR$(7);"    Caution - ";f1$;" already ";
10340 IF M1>0 THEN LINE INPUT "exists.  Do you want to revise it (Y/N)<N>? ";A9$
10350 IF LEFT$(A9$,1)<>"Y"THEN 11150
10360 PRINT :OPEN F1$ FOR WRITE AS FILE #1
10370 PRINT "    Enter the recipe lines following the '?' Enter 'END' to stop."
10380 Q=1:PRINT
10390 PRINT "Line ";Q;"";
10400 LINE INPUT ;L$
10410 IF L$="END"THEN 10450
10420 PRINT #1,L$
10430 Q=Q+1
10440 GOTO 10390
10450 CLOSE #1
10460 PRINT :PRINT "    This completes the '";F1$;"' recipe entry."
10470 IF M1>0 THEN 10490:REM No addition to the directory DISKCAT.DAT file
10480 GOSUB 10870:REM Delete this line when creating 'DISKCAT.DAT'
10490 PAUSE 750:GOTO 11150
```

```
10500 :
10510 REM Search routine
10520 :
10530 CLEAR :PRINT :LINE INPUT "What recipe file do you want?  ";A$
10540 PRINT :GOSUB 11030:REM Check to see if it exists
10550 IF M1=0THEN PRINT A$;" is not a valid file name.":PRINT :GOTO 11160
10560 OPEN A$ FOR READ AS FILE #1
10570 DIM D$(500):PRINT CHR$(27);CHR$(69):REM Increase for larger files
10580 GOSUB 11190:REM Get date
10590 PRINT :PRINT A$;" recipe file contents as of ";D1$;":":PRINT
10600 J=CIN(1):REM Read character from open channel; if negative, EOF was read.
10610 IF J<=0THEN 10660
10620 LINE INPUT #1,;S$
10630 S$=CHR$(J)+S$
10640 PRINT S$:REM May also be directed to any other device, like a printer
10650 GOTO 10600:REM Get another character
10660 CLOSE #1:PRINT :PRINT "(Press RETURN to continue)":PAUSE
10670 PRINT CHR$(27);"E":LINE INPUT "Do you want another recipe (Y/N)<N>? ";A$
10680 IF LEFT$(A$,1)="Y"THEN 10530
10690 GOTO 11150
10700 :
10710 REM Print out the contents of directory catalog
10720 :
10730 GOSUB 11190:REM Get date
10740 PRINT "    Recipe catalog as of ";D1$;":":PRINT
10750 OPEN "DISKCAT.DAT"FOR READ AS FILE #1
10760 J=CIN(1)
10770 IF J<=0THEN 10820
10780 LINE INPUT #1,;S$
10790 S$=CHR$(J)+S$
10800 PRINT S$;"  ":PRINT
10810 GOTO 10760
10820 CLOSE #1
10830 GOTO 11160
10840 :
10850 REM Update catalog directory
10860 :
10870 OPEN "DISKCAT.DAT"FOR READ AS FILE #1
10880 J=CIN(1)
10890 IF J<=0THEN 10930
10900 LINE INPUT #1,;S$
10910 S$=CHR$(J)+S$
10920 GOTO 10880
10930 CLOSE #1
10940 OPEN "DISKCAT.DAT"FOR WRITE AS FILE #1
10950 S$=S$+"/ "+F1$
10960 PRINT #1,S$
10970 CLOSE #1
10980 PRINT "    Recipe file updated."
10990 RETURN
11000 :
11010 REM Match new name with existing names of files
11020 :
11030 OPEN "DISKCAT.DAT"FOR READ AS FILE #1
11040 J=CIN(1)
11050 IF J<=0THEN 11090
11060 LINE INPUT #1,;S$
11070 S$=CHR$(J)+S$
```

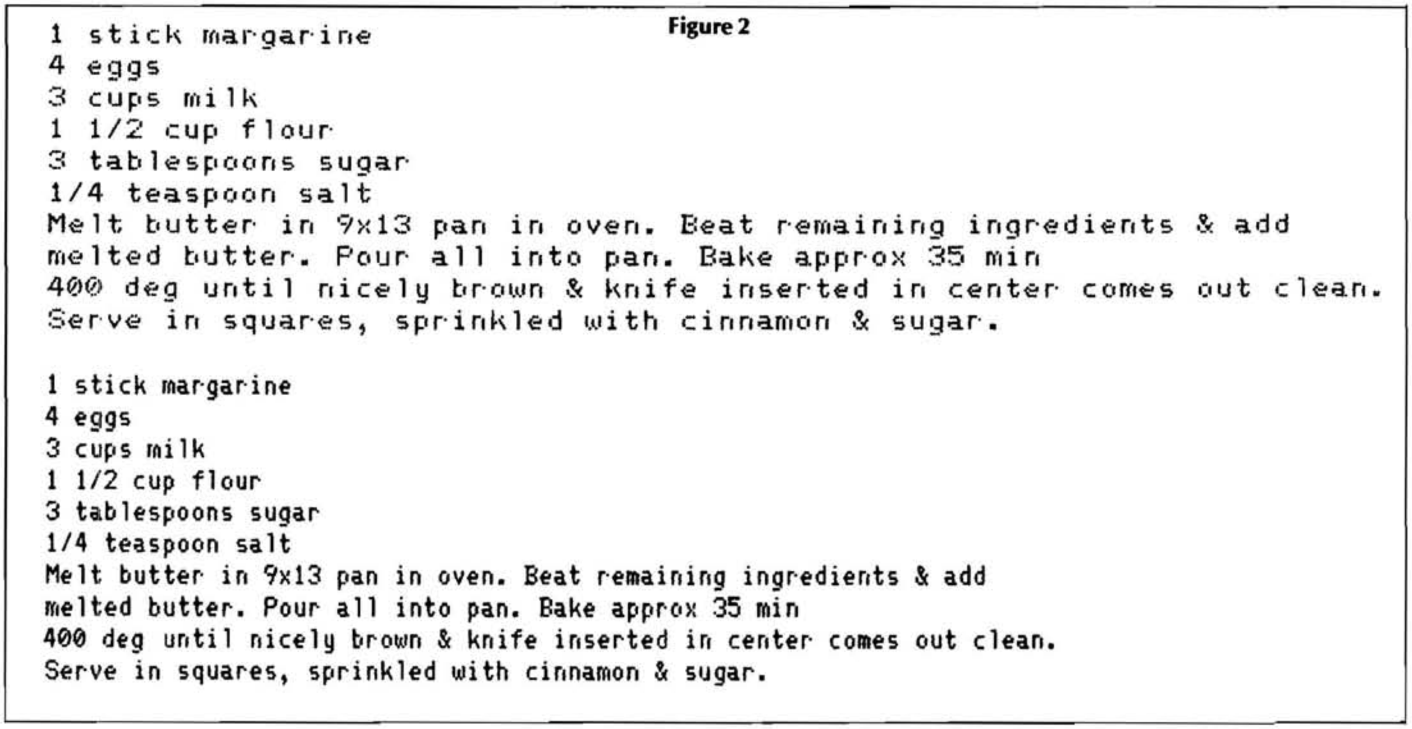
```
11080 GOTO 11040
11090 REM :Find occurance of A$ in S$, return index of A$ if found, 0 if not
11100 CLOSE #1:M1=MATCH(S$,A$,1)
11110 RETURN
11120 :
11130 REM Return to the program menu
11140 :
11150 PRINT CHR$(27);CHR$(69)
11160 PRINT :PRINT "Program menu:":PRINT :GOTO 10120
11170 :
11180 REM Date routine
11190 D1$="":FOR I1=8383 TO 8391:D1$=D1$+CHR$(PEEK(I1)):NEXT I1
11200 RETURN
```

Figure 1



```
2 pkg dry yeast
1/2 cup warm water (105-115 deg)
1 1/4 cups buttermilk
2 eggs
5 1/2 cups Gold Medal flour*
1/2 cup softened butter or margarine
1/2 cup sugar
2 teaspoons baking powder
2 teaspoons salt
Dissolve yeast in water in large mixing bowl. Add buttermilk,eggs,
2 1/2 cups of flour,butter,sugar,baking powder & salt. Beat 2 min
@ med speed. Stir in remain 3 cups flour. (Dough should remain soft
@ slightly sticky) Knead 5 min or about 200 turns on lightly floured
board. Shape as desired. Let rise in warm (85 deg) place until
double, about 1 hr. (Dough is ready to bake if slight dent remains
when touched) Heat oven to 375 deg. Bake until golden brown, about
45 min.
*When using self-rising flour, omit baking powder & salt.
```

Figure 2



```
1 stick margarine
4 eggs
3 cups milk
1 1/2 cup flour
3 tablespoons sugar
1/4 teaspoon salt
Melt butter in 9x13 pan in oven. Beat remaining ingredients & add
melted butter. Pour all into pan. Bake approx 35 min
400 deg until nicely brown & knife inserted in center comes out clean.
Serve in squares, sprinkled with cinnamon & sugar.
```

```
1 stick margarine
4 eggs
3 cups milk
1 1/2 cup flour
3 tablespoons sugar
1/4 teaspoon salt
Melt butter in 9x13 pan in oven. Beat remaining ingredients & add
melted butter. Pour all into pan. Bake approx 35 min
400 deg until nicely brown & knife inserted in center comes out clean.
Serve in squares, sprinkled with cinnamon & sugar.
```

The Heath/Zenith 88-89 CPU with a Future

SUPER 89

The Super 89 replaces the central processor board in the Heath/Zenith 88-89 series computers to bring your 88-89 to current state-of-the-art technology. The Super 89 gives you features that are useful today and allow expansion of your capabilities. The Super 89 is fully compatible with all Heath/Zenith products and also supports many peripheral devices from other manufacturers. New software and hardware are enhanced with the Super 89 by using all the features of the Z80 technology.

The Highlights of the Super 89:

- Twice the operating speed (4MHz +)
- Memory Capacity to 256 K in Software Bank Selectable 64K blocks
- CP/M and HDOS Compatible without modification
- Twice the number of expansion slots (Six)
- Real time clock on-board
- Two serial I/O Ports
- Designed for multi-user capability
- Parity checking for RAM assures integrity of memory transfer operations
- Arithmetic processor provision facilitates mathematic operations

These features, along with an enhanced monitor to access to all the Z80 CPU, give you power from your 88-89 that only large computers can claim.

High Speed Processing

The Super 89 runs twice as fast as the standard H/Z CPU board. Time savings on running programs are significant.

Expanded Memory Capacity

This feature allows you to use the advantages of the more sophisticated programming languages; enables you to use enhanced memory software such as print spoolers and electronic disks to increase speed; allows the use of "scratch pad" memory to increase efficiency; the bank select features give you high speed data handling and manipulation; and provide for multi user capabilities.

Super 89 Electronic Disk

This optional software package for the Heath/Zenith CP/M 2.2.02 and 2.2.03 allows the Super 89 user to access auxiliary RAM as a very fast mass storage device. Provides up to 180K bytes of storage area that is accessable without the slowness of disk drives. The Electronic Disk also includes display capability for the Real Time Clock.

Peripheral Expansion

This important feature lets you use your Super 89 in more ways with peripherals from DG, Heath and many other manufacturers. Some of these important enhancements are: Additional floppy disk controllers; Modem or Printer serial interfaces; Color video controllers; IEEE 488 BUS for test equipment and measuring interface; Analog/Digital interface; Parallel interface for high speed printers; Hard disk system controllers; Bread-board development cards; Computer game controllers; and Production process controllers.

Enhanced Super 89 Monitor

Gives you all the features of Heath's MTR-89 monitor plus the ability to display all the Z80 register contents; Single-step through a program and set up break-points; Supports H/Z and other manufacturers of disk systems; Improved system diagnostic routines; and Supports the Super 89 Real Time Clock.

Real Time Clock

The Real Time Clock allows you to program activities and control functions according to time; allows the use of interactive time functions with an electronic disk; and is very useful in accounting functions.

Parity Checking

This features ensures the integrity of memory transfer operations. The Super 89 alerts you if a parity error occurs.

Full CP/M and HDOS Compatibility

The Super 89 has full compatibility with either the HDOS or CP/M disk operating systems that does not require hardware modifications. This feature gives you the best of both worlds in the amount of existing software you may use.

Arithmetic Processor Provision

The Super 89 has on-board provisions for the optional AM9511A. This is a separate processor that features basic arithmetic as well as exponential, logarithmic, trigonometric and binary functions. Calculations are high speed and can be accomplished as a "hardware subroutine". This device is a must for anyone using any amount of mathematical computation whether complex functions or arithmetic calculations.

Ease of Installation

The Super 89 is simple to install and takes only minutes. No soldering required. Simply remove the old CPU board, configure and install the Super 89 to multiply the capabilities of your H/Z 88-89.

D·G ELECTRONIC DEVELOPMENTS CO.

Ordering Information: Products listed available from DG Electronic Developments Co., 700 South Armstrong, Denison, Tx. 75020. Check, Money Order, VISA or MasterCard accepted. Phone orders (charge only) call (214) 465-7805. Freight prepaid. Allow 3 weeks for personal checks to clear. Texas residents add 5%. Foreign orders add 30%. Prices subject to change without notice.

Heath®, H8®, H17®, and H89® are registered trademarks of Heath Corporation, Benton Harbor, Michigan. Z80® is the registered trademark of Zilog Corporation. AMD9511 is a trademark of Advanced Micro Devices.

ERROR IN OUTPUT FILE, FILE CLOSED...

Don't Despair, Help is on the Way

Robert H. French
1919 Rochelle Ave., Apt. 1831
District Heights, MD 20747

EDITOR'S NOTE: The program AUTOSCRIBE mentioned in this article is available through the Heathkit Catalog as ***ZENITH ELECTRONIC TYPING*** part number **H-8-40**

If you use the AUTOSCRIBE Word Processor sold by Heath, you are probably all too familiar with the message "AUTOSCRIBE ERROR HANDLER 4.0 ERROR IN OUTPUT FILE, FILE CLOSED. ENTER ANY CHARACTER TO ACKNOWLEDGE." After spending three hours revising a document, only to have the output disk too full to hold the revision would make anyone feel bad. But then when you display the index of the "full" disk and see 30,000 characters free..., well crying or swearing at the blinkety-blank system would not seem inappropriate. Don't despair, though, for you may be able to recover your hours of toil!

The key to the possibility of recovery is the fact that AUTOSCRIBE from time to time will save a temporary version of the file you are revising. This temporary version apparently is created whenever the document scrolls past the size of the work buffer kept in memory by AUTOSCRIBE. This temporary file is not deleted until the final version is written to the disk. Thus it is taking up valuable disk space, resulting in the error message. After the final output file is closed, this temporary file gets deleted. But wait! If this temporary file can be UNDELETED(!) and AUTOSCRIBE made to know about it, you can recover much of your work, up to the point at which the last temporary file was created. This means most, if not all, of your work can be potentially recovered.

To effect the recovery, you will need three things besides AUTOSCRIBE: 1) HDOS, Version 1.6 or higher; 2) an absolute file dump/patch program such as DUMP from HUG Disk VIII; and 3) scratch pad and pencil. The file dump/patch program must have the ability to do absolute track/sector access on an UMOUNTED disk. You will need at least two drives to use the procedure outlined below.

The steps outlined below will take you from the point at which the error message was given by AUTOSCRIBE through the recovery of the file. Follow the steps carefully, for if you make an error in the wrong place the entire disk's contents can be lost forever. If you don't feel confident of not making an irrecoverable mistake, you can use DUP from the same HUG disk to make an exact copy of the "full" disk prior to attempting the recovery. This will serve as a backup to any other files on that disk.

1. After you have finished lamenting your problems, press a space (or any other character) to acknowledge the error message and get back to the STARTING MENU.

2. Display the document index of the "full" disk. Take note of the first (lowest numbered) entry showing "No Document" as the name. This is the location into which you will wind up inserting a document named RECOVERY which will hold your recovered file.

3. Return to the STARTING MENU.

4. Remove the disk(s) from the drives and reset the computer.

5. Boot up from HDOS. The disk should also contain the DUMP program from HUG disk VIII. If you have three drives, you can alternatively boot from any HDOS disk and mount in drive SY2: the disk containing DUMP. In this case, be sure to type SY2:DUMP to run dump.

6. Mount the "full" AUTOSCRIBE disk in SY1: by placing the disk into the drive (known also as AUTOSCRIBE drive 2) and typing the command MOUNT SY1: in response to the › prompt from HDOS.

7. Use DUMP in the file mode as described in the documentation which comes with the program to read the 1-sector file named GRT.SYS on the full AUTOSCRIBE disk. The command is:

```
>DUMP SY1:GRT.SYS
```

8. Note down the following information from the display on the screen:

a. The contents of the first byte of the file (a two-digit hexadecimal number).

b. The Track and Sector at which the information is located on the disk (given at the top of the DUMP display).

9. Use DUMP in the file mode to examine the file DIRECT.SYS on the full AUTOSCRIBE disk. You will have to use CTL-S/CTL-Q to stop and start the scrolling of this multiple-sector file. Watch for two entries in the directory: ADIR.DOC and a file which will show up in the ASCII interpretation on the right of the screen as OC.TMP. The actual names in the directory entries will contain blanks padding the name to a full 8 characters, except for OC.TMP which, being a deleted file, will have a non-printable hex FF substituted for the first character. YOU MAY FIND MORE THAN ONE "OC.TMP" IN THE DIRECTORY! The one you want will have the 17th byte (the FF byte before OC is the first byte, etc.) equal to the first byte of GRT.SYS which you noted down in step 8a above. Note down the track and sector of DIRECT.SYS in which the entries for ADIR.DOC and OC.TMP are found. Also note down the 17th and 18th bytes of the entry for ADIR.DOC.

10. Divide the number of the first free document position in the AUTOSCRIBE directory (the "No Document" entry noted earlier) by 4, dropping the fraction. The result should be a number in the range 0 to 9. If the result is 0, note the number from the 17th byte of the ADIR.DOC entry in DIRECT.SYS as the group number of the ADIR entry you want and go to step 13.

11. If the result of the division in step 10 was 9, note the number from the 18th byte of the ADIR.DOC entry as the group number of the ADIR entry you want and go to step 13.

12. If the result of the division in step 10 was neither 0 nor 9, do the following:

a. Use DUMP to display the file GRT.SYS on the "full" AUTOSCRIBE disk.

b. Locate the byte of the file whose number equals the contents of the 17th byte of the entry for ADIR.DOC in DIRECT.SYS.

c. Note the contents of the byte located in 12b.

d. Using the number from step 12c in place of the contents of the 17th byte of ADIR.DOC, repeat steps 12b and 12c a total number of times equal to the result of the division in step 10. Note that if the result of that division was 1, no repetitions are done, and the first number located in the GRT.SYS is the number you want.

e. Note the final number located as the group number of the ADIR entry you want.

13. If you are still in DUMP, CTL-C back to HDOS.

14. DISMOUNT THE "FULL" AUTOSCRIBE DISK! The command is:

```
>DISMOUNT SY1:
```

This step is very important; if omitted, things will NOT work properly in the next steps.

15. Using the absolute track and sector numbers noted down earlier, use DUMP in the absolute mode to access the GRT.SYS file on the AUTOSCRIBE disk. Tell DUMP "Y" when it asks if the sector is to be modified.

16. Change the first byte of GRT.SYS to 00 (hex) and write it back to the disk.

17. Access, by absolute track and sector number noted earlier, the sector of DIRECT.SYS containing the deleted file OC.TMP. Change the FF byte to 41 (hex). Change the next two bytes to the ASCII equivalents of the free document number noted down from the AUTOSCRIBE directory. The ASCII (hex) equivalent of a decimal digit is 3 followed by the decimal digit. Don't forget the leading zero in the range of document numbers 01-09! Leave the next five characters as they are, then change the three bytes (which should contain the hex codes 54 4D 50) to 44 4F 43. Write the sector back to the disk.

18. Convent the hexadecimal group number for the ADIR entry obtained in step 10, 11, or 12 to decimal. Multiply the result by 2. Divide the result by 10. The integer part of this number is the track number and the decimal part is the sector number within that track to access the sector of ADIR which you need to change.

19. Using the track and sector numbers calculated in step 18, use DUMP to access the sector of ADIR.DOC containing the information on the first unused entry which was noted at the beginning of this procedure. This sector will contain the information on 4 documents. The first byte will contain, in hexadecimal, the AUTOSCRIBE document number. Each entry occupies 4 lines of the DUMP display. Locate the group of four lines for the document number noted as being a "No Document" entry. The third byte of the first line should be 00 (hex). Change this byte to 01 (hex). Beginning at the 7th byte of the same line (if the line is address x0, the beginning address is x6), change 24 consecutive bytes to read: 52 45 43 4F 56 45 52 59 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20. Change the first 11 (decimal) bytes of the third line of the entry, i.e. the one with address (x+2)0, to hexadecimal 00's. Write the sector back to the disk. You may find it easier to make one group of changes, write the sector back to the disk, and then access it again to make the next group of changes, rather than try skipping the intervening bytes.

20. Hit CTL-C to get back to HDOS.

21. Type the command

```
>BYE
```

to HDOS to allow you to reboot.

22. Reboot from AUTOSCRIBE System Disk.

23. Place the disk you have been altering (if not the System Disk) into Autoscribe Drive 2 or 3. Display the index, and you should find a document called RECOVERY in the directory with no date and 0 bytes in it. This should contain your recovered file! View it to see how much is there, and then take up where you left off, backing up as necessary to incorporate any of your last few edits/additions which were lost. BUT...be sure you have lots of free space on the output disk, unless you want to practice this procedure again!

In summary, what the above has accomplished is to undelete the document DOC.TMP, rename it Axx.DOC (where xx is a free AUTOSCRIBE document number), and incorporate into that file all free space on the disk. Since the disk was full before this file was deleted, the available free space on the disk should be exactly what was in the file before it was deleted. This undeletion operation requires fiddling around with the files DIRECT.SYS and GRT.SYS on the AUTOSCRIBE disk to undelete the file within the context of the (modified) HDOS under which AUTOSCRIBE runs. Then the AUTOSCRIBE directory, which is kept in file ADIR.DOC, is patched up to make the word processor aware of the file, and give it the name RECOVERY. Rather than fiddle around with an ASCII table to insert a good date, the date is zeroed (making it look like a file from the older versions of AUTOSCRIBE which did not enter a date) and the unknown byte count is set to zero. Neither of these should affect the usability of the document for further revisions.

If you never, never create large documents and ALWAYS keep lots of free space on your document disks, you will probably never need the above process. But, if you are like most people, you will likely need this some day. Good luck and happy word processing!

Base Conversion Utility for MBASIC

Vince Saunders
155 Mont Bleu #211
Hull, Quebec, Canada
J8Z 1K5

Have you ever been in the process of writing a program in MBASIC and needed to obtain the octal equivalent of a Hex number, or decimal value to put into a function to obtain some specific result? Usually when this happens it is necessary to save your partially completed program, exit BASIC to System and maybe even boot up some other disk to be able to load CONVERT.ABS or CALCUL.ABS or some other conversion routine. Then you entered your values, obtained the conversions, wrote them down on some scrap of paper, dismounted that disk, mounted your BASIC disk, loaded MBASIC, loaded your program and then couldn't find the scrap of paper with your answers on it!

This small program will allow you to do conversions in Decimal, Octal and Hexadecimal while still in MBASIC and with your program still loaded and you can even run your program to check results.

The program is written in MBASIC and utilizes the intrinsic MBASIC function Oct$(x), Hex$(x), $Hxxx and &0xxx to perform rapid conversions. The program runs from line 65000 to line 65056 and should be saved in ASCII format on your MBASIC disk. When needed the program is merged into your program. Then you should insert a line '64999 END' to prevent your program from running into the conversion routine and developing errors such as RETURN without GOSUB.

To use the program when in command mode enter GOTO 65000 and the conversion program will run and prompt you for input. To return to your program type 'E' to quit and you will be in command mode.

To avoid cross-talk or side-effects from one program to another avoid using variables in your program with the same name as in the conversion program. A cross reference listing is supplied to assist you in avoiding clashes. To further prevent trouble all variables in CONVERS.BAS are double letter variable names such as BB$ except for the variable name CHAR$.

Program explanation:

lines 65006 to 65014

These lines issue the prompt for base type, test for upper and lower case conditions of the input, return to prompt if invalid input and drop through for case decimal. Sets 'N' to integer.

lines 65015 to 65017

Line 65015 accepts input string, sub-routine at 65047 tests string for invalid characters, takes care of minus sign in decimal conversion and returns. Any error sends the program back for new input. Line 65017 sets 'NN' equal to the integer value of the string NN$.

lines 65018 to 65021

These lines simply print the results using the MBASIC intrinsics and holds display until a RETURN is pressed and returns the program to the prompt.

lines 65023 to 65030

These lines perform the same functions as for the decimal case except that routine 65047 now has the additional task of testing for valid input of A,B,C,D,E and F as well as catering for the lower case input of these characters. Line 65049 extracts the character, tests if greater than 91 decimal, the upper decimal limit of the upper case characters, and does an XOR with octal 040 to convert the lower case character to upper case. This conversion is only done in this routine so that the original input is printed back in the results.

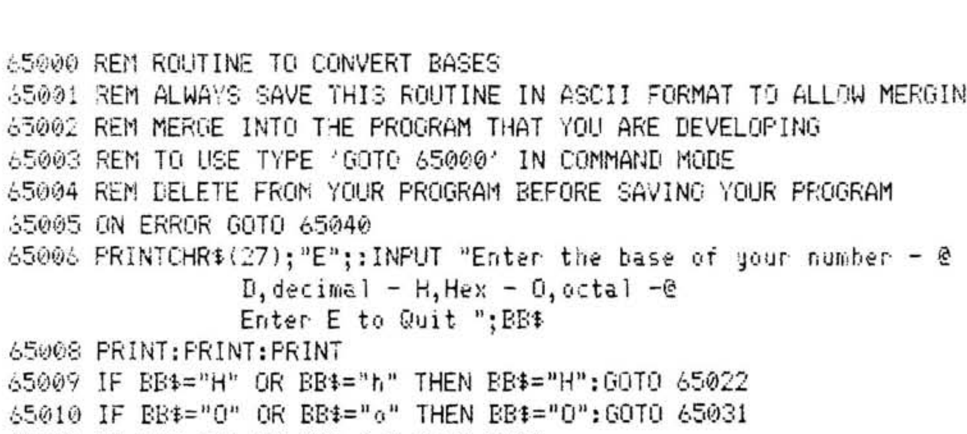
lines 65032 to 65039

Here we perform the same operations except that the octal input is limited to numbers from 0 to 7 only. This test is carried out in line 65054.

lines 65040 to 65046

This is the error trapping routine which traps the MBASIC error 'OVERFLOW' and converts it to a 'Number loo large' message and resumes execution at line 65006 the prompt.

There are no Remarks within the active program in order to avoid loss of speed caused by MBASIC having to read Remark statements that do not contribute to program execution and to make the program as small as possible. Once you are familiar with the program I suggest that you remove all Remarks and renumber line 65005 or you may renumber the entire program as 65006,65000,1.



```
65000 REM ROUTINE TO CONVERT BASES
65001 REM ALWAYS SAVE THIS ROUTINE IN ASCII FORMAT TO ALLOW MERGING
65002 REM MERGE INTO THE PROGRAM THAT YOU ARE DEVELOPING
65003 REM TO USE TYPE 'GOTO 65000' IN COMMAND MODE
65004 REM DELETE FROM YOUR PROGRAM BEFORE SAVING YOUR PROGRAM
65005 ON ERROR GOTO 65040
65006 PRINTCHR$(27);"E";:INPUT "Enter the base of your number - @
                D,decimal - H,Hex - O,octal -@
                Enter E to Quit ";BB$
65008 PRINT:PRINT:PRINT
65009 IF BB$="H" OR BB$="h" THEN BB$="H":GOTO 65022
65010 IF BB$="O" OR BB$="o" THEN BB$="O":GOTO 65031
65011 IF BB$="E" OR BB$="e" THEN STOP
65012 IF BB$<>"D" AND BB$<>"d" THEN 65006
65013 BB$="D"
```

```
65014 DEFINT N
65015 INPUT" Enter the decimal number you wish converted ~ ";NN$
65016 GOSUB 65047
65017 NN=VAL(NN$)
65018 PRINT:PRINT:PRINT
65019 PRINT"    Decimal = ";NN;"     Hex = ";HEX$(NN);"     Octal = ";OCT$(NN)
65020 PRINT:PRINT:PRINT:INPUT "Press RETURN to Continue ";ZZ$
65021 GOTO 65006
65022 PRINT:PRINT:PRINT
65023 INPUT " Enter the Hexadecimal number you wish converted ~ ";NN$
65024 GOSUB 65047
65025 HH$="&H"+NN$
65026 PRINT:PRINT:PRINT
65027 DD=VAL(HH$)
65028 PRINT "    Decimal = ";DD;"     Hex = ";NN$;"      Octal = ";OCT$(DD)
65029 PRINT:PRINT:PRINT:INPUT "Press RETURN to Continue ";ZZ$
65030 GOTO 65006
65031 PRINT:PRINT:PRINT
65032 INPUT " Enter the Octal number you wish converted ~ ";NN$
65033 GOSUB 65047
65034 OO$="&O"+NN$
65035 OO=VAL(OO$)
65036 PRINT:PRINT:PRINT
65037 PRINT "    Decimal  = ";OO;" Hex   = ";HEX$(OO);"   Octal   = ";NN$
65038 PRINT:PRINT;PRINT:INPUT "Press RETURN to Continue ";ZZ$
65039 GOTO 65006
65040 IF ERR=6 AND ERL = 63017 THEN 65044
65041 IF ERR=6 AND ERL = 65027 THEN 65044
65042 IF ERR=6 AND ERL = 65035 THEN 65044
```

```
65043 ON ERROR GOTO 0
65044 PRINT:PRINT:PRINT:PRINT"      Number too large :"
65045 FOR II=1 TO 1000:NEXT
65046 RESUME 65006
65047 FOR II=1 TO LEN(NN$)
65048 IF LEN(NN$)=0 THEN RETURN
65049 CHAR$=MID$(NN$,II,1):IF ASC(CHAR$)>91 THEN CHAR$=CHR$(ASC(CHAR$)XOR &O40)
65050 IF BB$="D" AND CHAR$="-" THEN 65055
65051 IF BB$="D" AND (ASC(CHAR$)<48 OR ASC(CHAR$)>57) THEN 65015
65052 IF BB$="H" AND (ASC(CHAR$)<48 OR (ASC(CHAR$)>57 AND ASC(CHAR$)<65)) @
        THEN 65023
65053 IF BB$="H" AND ASC(CHAR$)>70 THEN 65023
65054 IF BB$="O" AND (ASC(CHAR$)<48 OR ASC(CHAR$)>55) THEN 65032
65055 NEXT II
65056 RETURN
```

☞ ATTENTION: ☜

Local Clubs and Heath/ Zenith Vendors

In an effort to bring new members of the Heath Users' Group up to date in this fast paced world of personal computers, we are now beginning to compile the January 1983, Issue of REMark. The Heath Users' Group will be providing this issue as a cross- reference of all REMark articles, all Local HUGs that wish to participate, all vendors that desire mention in REMark, and finally, a brief description of the HUG software released during 1982.

We would like to supply the latest information on your club via the club officers and/or the latest that your company has to offer in the way of Heath or Zenith related hardware or software. If your club or company wishes to be listed, please check the following information for details.

Clubs:

Club Name:
Club Address:
Contact Individual: Phone Number:
Group Size:
Meeting Location: Meeting Time:
Bulletin Board: Yes No Phone Number:
Special Notes:

Heath/Zenith Related Vendors:

Company Name:
Company Address:
Contact Individual: Phone Number:
Hardware: Yes No Software: Yes No
Consultation Available: Yes No
Description of Product(s) (A total of 25 words or less for all products):
Special Notes:(Max. 15 words)

To accomplish our goal of providing the necessary information for our members in the January 1983, Issue of REMark, will require that we have your information no later than **December 1, 1982.** Please forward your information TYPED in a neat format that will allow for ease of entry.

For the Local Clubs: If the information appearing in the August 1982, Issue 31 of REMark is correct, we will not require further contact until changes are necessary.

Send your information to:
Attention: Walt Gillespie, REMark Editor
Heath Users' Group
Hilltop Road
St. Joseph, MI 49085

Local HUG groups

As you will see, more local HUG groups are forming and continued interest grows. Keep those inquiries on local HUG groups coming!

R. Sibers 94 Dowser Drive Maungaradi Lower Hutt New Zealand is interested in starting a local HUG club in NEW Zealand. Anyone interested should contact him.

Harold Dykens and seven other individuals have expressed a similar interest in starting a HUG group in Des Moines, Iowa. Currently they plan to meet on the third Wednesday of each month at 7:00 p.m. Anyone interested may contact Harold at 4025 E. 32nd St. Des Moines, IA 50317 or call (515)266-2382.

Stan Lockhart wants to start a HUG club in the Fort Greely/Fairbanks, Alaska area. Anyone interested may contact Stan at PO Box 229 Fort Greely, AK 98733 or call (907)895-3294.

Anyone interested in starting a HUG club in Nashville, Tennessee should contact Charles Q. Wolf at Radio Service Center 116 17th Ave S. Nashville, TN 37203 or call (615)242-0556.

New Groups:

JUG (Jacksonville Users Group)
8262 Arlington Expressway
Jacksonville, FL 32211
904 725-4554 Group size 40
Contact person: Jerry Leon
BB 725-4995 24 hours
Meet 1st Wed at HEC

San Jose HUG
2350 S Bascom Ave
Campbell, CA 95008
408 377-8472 Group size 70
Contact person: Gerlene York, Sec.
Meet 1st & 3rd Wed 7:00pm HEC

BIHUG (Big Island HUG)
PO Box 4271
Hilo, HI 96720
808 961-4846 Group size 10
Contact person: R.A. Curtis
Meetings at HELCO in Hilo as called.

ATTENTION USERS OF CP/M PAYROLL

....CORRECTIONS TO CP/M PAYROLL....

HUG PART NUMBER 885-1218

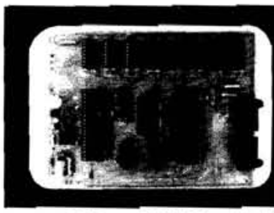
In the last minute rush to release the CP/M Payroll Package now offered by the Heath Users' Group, an incorrect file was transferred to the master disk used for duplication. The program PAYSTART.BAS contains a syntax error in line 150 and, we have fc ınd that an addition to line 125 is required to insure proper operation on a single drive system. Included below are the completed and corrected lines of the program. If you are attempting to use CP/M Payroll, please incorporate these changes.

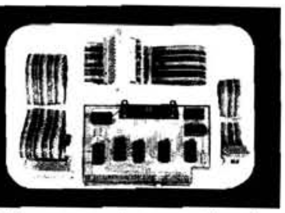
PAYSTART.BAS corrections:

```
125 A$=INPUT$(1):RESET:PRINT E$"x5":GOTO 40

150 IF ERL=70 AND ERR=33 THEN PRINT Y$"&0 This is not the
"P$" PROGRAM "Q$" DISK. ":RESUME 110
```

Thanks for your patience and cooperation.





As the first units begin to roll off the production line, Bill Johnson, (on the left) President of Heath Company seals up the carton of a new Z100 computer as Barry Watzman, Computer Product Line Manager, fills out paper work and Don Moffett, President of Zenith Data Systems (on the right) removes a test disk from a low profile Z100.

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

When was the last time you renewed?

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 ________________________________

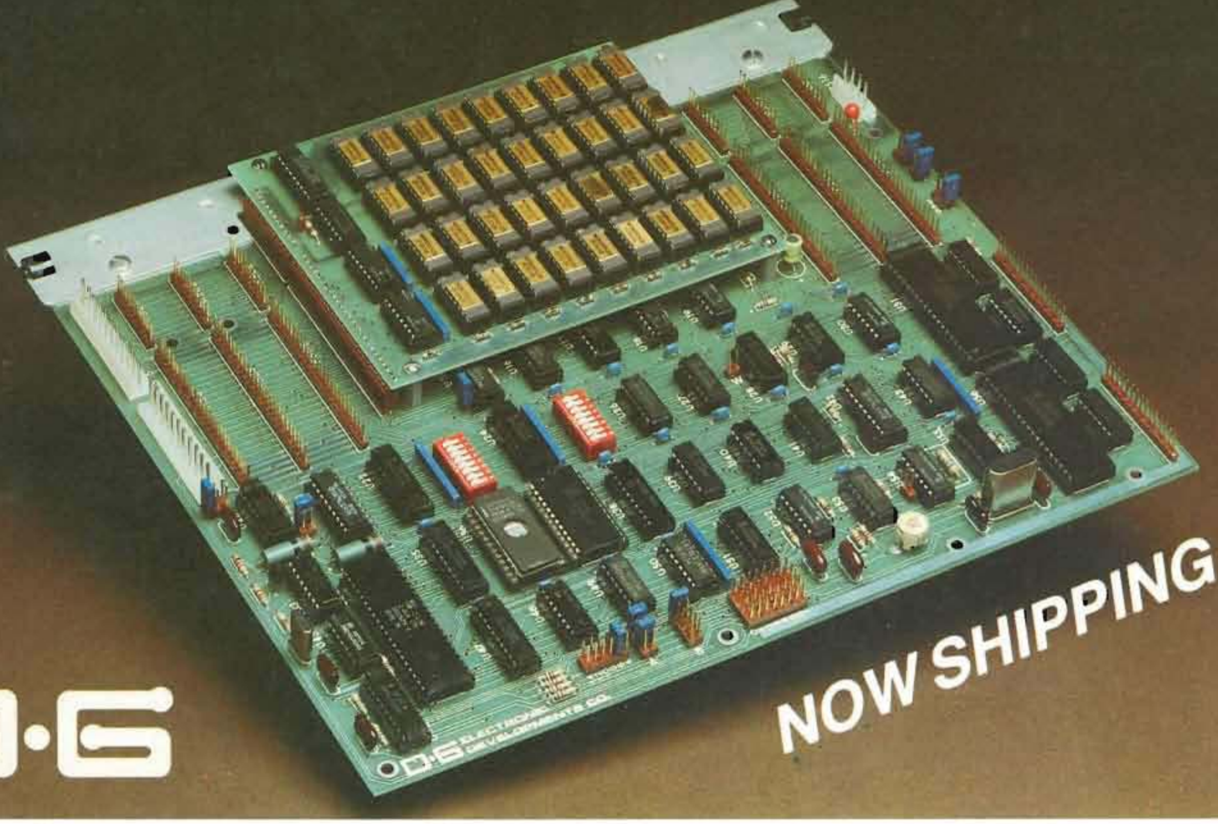
Zip ________________________________

REMEMBER — ENCLOSE CHECK OR MONEY ORDER

CHECK THE APPROPRIATE BOX AND RETURN TO HUG

		NEW MEMBERSHIP FEE IS:
RENEWAL RATES		
US DOMESTIC	$15 □	$18 □
CANADA	$17 □ US FUNDS	$20 □
INTERNAT'L*	$22 □ US FUNDS	$28 □

* Membership in England, France, Germany, Belgium, Holland, Sweden and Switzerland is acquired through the local distributor at the prevailing rate.



Heath
Users'
Group
Hilltop Road
St. Joseph, MI 49085