
$2.50
REMark®
Issue 42 • July 1983
HUGGERS PREPARE FOR THE 2ND NATIONAL HUG CONFERENCE
see page 5 for details
Official magazine for users of HEATH ZENITH computer equipment.

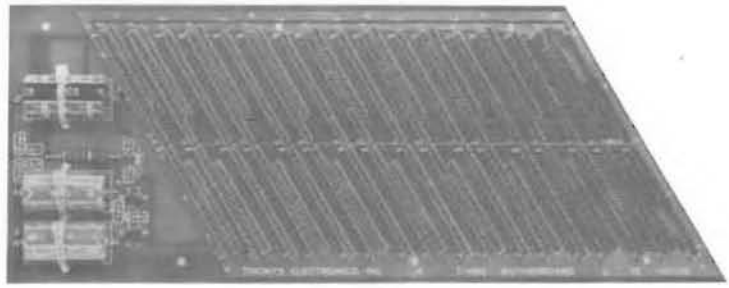
Introducing –

THE NEW H8* COMPUTER

* H8 is a Registered Trademark of the Heath Company

PC Board Mounts Additional Pair of Connectors to Plug Into 90 Pin Bus

Optional 40 Pin Connector Assembly Converts H8 Boards to 90 Pin Bus



Fully Assembled Motherboard - $250.00

Featuring the TRIONYX T-H90 MOTHERBOARD
A Professional-Quality Bus for the H8 Computer

- 3-Layer Board Has Center Ground Plane
- Designed For 4 MHZ Bus Operation
- 7 Auxillary Positions For Port Addressable Bus Interface Cards
- Completely Compatable With Original H8 Computer

- 90 Pin Bus Has:
  - Additional Ground Connections For Reliable Operation
  - 8 Additional Data Bits For 16 Bit CPU Board
  - Additional Address Lines For Expanded Memory Capacity
  - Bank Select Lines For Memory Management
  - Additional Lines For Special Control Signals and Future Defined Functions

PC Board $ 75.00

25 Pin Gold Connectors - Set of 20 $45.00
- Build Motherboard to Original H8 Standard

20 Pin Gold Connectors - Set of 18 $34.00
- Add Additional 40 Pins for 90 Pin Bus

25 Pin Gold Connectors - Set of 14 $35.00
- Add 7 Auxillary Card Positions

Power Supply Parts for Motherboard $16.00
- Includes Extra +18 Volt Filter Capacitor

Bus Termination Card - Complete Kit $29.50

PC Board Connector Expansion Kit $15.00
- Adapts Original H8 PC Boards to 90 Pin Bus: Contains PC Board, Two 20 Pin Connectors, Mounting Blocks and Hardware - One Kit Required Per PC Board

The new T-H90 Motherboard has been designed to provide completely reliable operation of the H8 computer through the use of gold-plated connectors and a well grounded bus. The bus has been expanded from 50 to 90 lines and 7 additional card slots have been added. Full implementation of the Motherboard functions will transform the H8 into a commercial grade computer.

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

Heath Users' Group

HUG Manager ..... Bob Ellerton
Software Engineer ..... Pat Swayne
HUG Bulletin Board
and Software Developer .... Terry Jensen
Software Coordinator ..... Nancy Strunk
HUG Secretary ..... Margaret Bacon

REMark Editor ..... Walt Gillespie
Assistant Editor ..... Donna Melland

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

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

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

*U.S. Funds.

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

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

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

616-982-3463

REMark®
Issue 42 • July 1983

on the stack

The Second National Hug Conference Approaches *Bob Ellerton* 5
Keyword-based Filing and Retrieval *Jim Tysinger* 13
A Z-100 Font Editor *Marc Aagenas* 19
Using the H/Z-29 Terminal *Pat Swayne* 25
The Wand vs. The Star *John Immerwahr* 29
Personal Checks and Balancing *Don Thomas* 33
Millionaire: A Stockmarket Simulator *Tom and David Huber* . . . 37
Getting Started With Assembly Language (#4) *Pat Swayne* . . . 40
A Simple Separate Keyboard for the H/Z-89 *Ernst S. Duesterhoeft* 43
"Changing Gears" in Your 4MHz H-89 *Peter Shkabara* 45
Using AUTOEXEC.BAT *Gerry Kabelman* 47
Addressing Envelopes/Labels *Charles 'Karl' Romer* 48
Programming Graphic Shapes *David E. Warnick* 55
Patch Page *Pat Swayne* 59
Draw and Print a Picture In MBASIC *Ron White* 61
Introduction to Z-BASIC Part VII *Gerry Kabelman* 63
BDOS ERROR *Bill Simpson* 65
Computer Aided Instruction Part 3 *Walt Gillespie* 68
Give HERO Something To Do With His Time *Pat Swayne* 71
Modem Tutorial 83, Part 2 *Walt Jung* 75
What Does SET SY: Set? *Bill Tavolga* 80

Schedule of Events 7
Discussion Group Schedule . 8
Conference Registration 9
Vendor/Booth Locations 10
Odds and Ends Sale 17
HUG New Products 50
HUG Price List 53
Update on Vendors 83
Current Local HUG Clubs . . 86
Questions and Answers 91
Buggin' HUG 92

ON THE COVER: Pictured is the National HUG Staff preparing to leave for the Second National HUG Conference in Chicago, Ill.

Welcome To The O'Hare Hyatt Regency

The Second National HUG Conference Approaches ...

Bob Ellerton
HUG Manager

It's that time again folks! With August rapidly drawing near, the final plans for the Second National HUG Conference are taking shape. We at HUG are busily collecting the loose ends, contacting speakers, preparing the Vendor Exhibit Area, and talking with the people at the O'Hare Hyatt Regency to ensure that the weekend you spend with other HUGgies is both entertaining and enjoyable. What have we cooked up for August 19, 20, and 21? Let's take a look!

The Vendor Exhibit Area

Probably one of the most enjoyable sections of the HUG National Conference is the Vendor Exhibit Area. This is the place where each of us gets a chance to look over the products and services offered by companies supporting the Heath/Zenith computers. Also, we might have the chance to chew-the-fat with individuals who we have met only through articles, networks, or phones. This year, the Vendor Exhibit Area is a real Who's Who of the Heath/Zenith family. With forty booths and forty-three exhibitors, we're sure you will find something to look at during the weekend.

Each of the vendors has been kind enough to donate a door prize (or they can't eat supper). Most of the door prizes this year will be given away at the Vendor Exhibit Area. Every two hours, six numbers will be selected at random from a file of attendees by computer. These numbers will be placed on a screen at the Heath Users' Group Booth. To claim a prize, you must bring your ticket with the matching number to the HUG Booth. If a door prize is not claimed within the two hour period, another six numbers are selected and displayed on the screen. Be sure to stop by the HUG Booth for details.

Of course, the Heathkit Electronic Centers will be represented again this year. Rumor (from a very reliable source) suggests that you might bring "a lotta green or a lotta plastic". In other words, be prepared for some excellent buys on software and hardware. As a warm-up to the National HUG Conference, Heathkit Electronic Centers will be offering "odds-and-ends" hardware items at greatly reduced prices (see information elsewhere in this issue). Items that are not sold locally will be available at the Conference along with other goodies that the Centers have selected as "Conference Specials".

Directly in front of the Vendor Exhibit Area, HUG will have three large bulletin boards available for the following purposes:

Wanted/Trade - This bulletin board will be available for those of you wishing to purchase or trade items.

For Sale - This bulletin board will be available for those of you wishing to sell computer related hardware or software.

Messages - This bulletin board will be available for general messages of any nature.

These bulletin boards are reserved for the users. Special forms will be available for these boards. We ask that the ads on the boards are for personal equipment only and that the ad only be removed by the individual who placed it there.

The booth arrangements are included with this article so you can be familiar with the vendors attending and their general location. The final program of events, along with any changes, will be available at the Registration Booth. Be sure to sign in even if you have received your tickets in advance from HUG. The Vendor Exhibit Area will be open to the users at 4:00 p.m. Friday, August 19, 1983.

What is a NUA Lounge?

On a final suggestion from a "computer widow", a NUA Lounge will be established for the weekend. What's a NUA? A NUA is a Non-User-Attendee. The NUA Lounge will be a place to relax your tired feet. HUG will have coffee, pop, etc. available in this area for the same hours that the exhibits are open. Twice during the Conference, a short meeting will be conducted for the beginner or the curious to become familiar with the fascinating world of micro-computers (see schedule for details).

Dropping in on the NUA Lounge will be a good idea for all of us.

Not only will this area be a place to relax, the Lounge will be a place to gain access to some great public-domain software. Bob Todd has agreed to set up a copying facility for his vast collection of "public" disks. Bring your blanks as Bob suggests that he will be able to handle all of the typical formats (no paper tape please).

Also, the NUA Lounge will have a large map of the United States to display the locations of the Local HUG Clubs that we have listed in this issue. Any additions/corrections to this list may be requested at the Conference by contacting one of the HUG Staff at the Registration Booth or HUG Booth.

It is our hope that each of you will enjoy the NUA Lounge as a new addition to the Second National HUG Conference. Plan on stopping in for a "sit" or a little conversation.

Friday night at the Second National HUG Conference

The official opening for the Second National HUG Conference is scheduled for Friday evening. At 8:00 p.m., a cash bar will be available in the Rosemont Ballroom. Ceremonies are scheduled for 9:00 p.m. After a brief introduction of the National HUG Staff members, special awards will be presented to selected individuals and groups.

These awards are presented for special contribution to the Heath/Zenith user community.

Following the awards presentation, our group will be hearing from Pat McNamara, Sales Manager, Zenith Data Systems. Pat has been asked to give a short presentation on the future direction of the Heath/Zenith computer product line. Without giving away his presentation, Pat addresses some of the more difficult questions you, as concerned users, have voiced in the past.

Moving on to Saturday

Saturday morning brings out the speakers for the Second National HUG Conference. The Conference schedule includes 24 discussion groups on eight topics. The rotating nature of the discussions will allow you to attend six of the possible eight topics. Careful planning, using the schedule included with this article, will enable you to make the most of your time during the weekend. As a suggestion, make at least two schedules since a session may fill up quickly depending on interest. Clubs who have more than one member in attendance may wish to record a session or take careful notes to be distributed to other attendees or club members. Since there will be three possible times to hear a particular discussion or speaker, please use your good judgement when a group fills.

Activities throughout Saturday include the Vendor Exhibit Area, the Discussion Groups, and the Heathkit Electronic Center. These activities, of course, are the warm up to the Second National HUG Conference Dinner.

Dinner Highlights

If you were fortunate enough to attend the First National HUG Conference, then you are aware of the efficiency of the O'Hare Hyatt Regency staff. A cash bar and reception area will be available in the United A and United B rooms Saturday evening at 6:30 p.m. At 7:30 p.m., dinner will be served.

After our dinner, we will again be privileged to hear from Mr. Bill Johnson, President of the Heath Company. Bill will be delivering the Keynote Address.

The most exciting event of the First National HUG Dinner was the drawing for a "Z" Machine, an H-89 computer and the H-25 printer. What's up this year? All indications would point to even more excitement this year.

Bill Johnson has indicated that he's going to be giving away a "pet" that you can build yourself. HMMMMM? Hey Bill!? Does it vac the RUG?

Pictured are three HUG-hers modeling the T- Shirts that will be available only at the 2nd National HUG Conference at $6.50 each.

How about Joe Schulte, President of VEC (Heathkit Electronic Centers)? Joe told HUG that he had a thing with an "H" that looks like a machine with a "Z". Any more clues, Joe?

HUG contacted Don Moffet, President of Zenith Data Systems. What did Don say? Well, Don was on his way to lunch (it's good to catch him at a weak moment), and after about two seconds he suggested that a 30-30 rifle might be appropriate for some lucky HUGgie. Don does it again!

Of course, HUG has some other goodies planned for the evening that we are sure will be of interest to all who plan to attend. As last year, you must be present to win any of the prizes available on Saturday evening. So, don't plan on a Big Mac or another party this year!

As we retire for the evening, plan on visiting with some of your HUG friends around the various facilities provided within the Hyatt. Last year, the HUG staff had a great time talking with, and meeting HUGgies from all over the country. It's nice to attach names with faces as the Conference progresses.

Sunday Overview

Sunday morning, Discussion Groups and the Vendor Exhibit Area open early to ensure that you have a chance to grab that last little piece of information or a last look at the products offered for the Heath/Zenith computer products.

Sunday afternoon features a HUG Club gathering in the Rosemont Ballroom with a presentation from one of our West Coast groups. Last year, we had the pleasure of hearing from Bill Johnson of the Capital Heath Users' Group. Bill gave us a well organized talk on the trials of establishing a local group. This year we will hear from the West Coast. It is hoped that the viewpoints presented at this important gathering will help other clubs just getting started or aid existing clubs to become more active.

HUG is preparing some information that will be presented at this gathering and is designed for club participation. The nature of this information will benefit the exchange of knowledge throughout the HUG community. Further, Bob Todd has agreed to share his knowl-

edge on gaining access to various public-domain software libraries. We hope that each of the clubs will have a representative available for this important session.

The Second National HUG Conference in Review

As you can see, the weekend at the Second National HUG Conference is going to be packed with activity. The outstanding backing of the Heath Company, VEC, and Zenith Data Systems will, to say the least, contribute to a very exciting Saturday evening.

We at HUG wish to extend our thanks to the vendors participating in the Second National HUG Conference as well as the vendors that continue to advertise in REMark. Without this group of loyal Heath/Zenith computer enthusiasts, there would be no possibility of a conference the likes of the National HUG Conference.

NOTE: The schedule of vendors and events for the Second National HUG Conference may be modified from the contents of this article. Changes to the schedule or Vendor Exhibit Area will be available at the Registration Booth. Changes that may occur during the Conference will be announced within the Vendor Exhibit Area. Tickets that have been purchased in advance are now being mailed every two weeks. If you have not received your tickets and it has been more than four weeks, please contact the Heath Users' Group for verification of your reservation to the Saturday night dinner party. Those of you planning to attend for one day may purchase tickets at the Registration Booth. Remember, however, tickets purchased for one day do not include dinner. You may win door prizes available during the day in the Vendor Exhibit Area. You are not eligible for prizes selected at the dinner unless you have purchased a ticket for, and attend, the dinner celebration.

Second National HUG Conference
Schedule of Events

Friday, August 19, 1983

1:00 pm	Registration Booth Opens
3:00 pm	Vendor Exhibit Area Opens
7:00 pm	Vendor Exhibit Area Closed
8:00 pm	Registration Booth CLosed
8:00 pm	Grand Opening Warm-up (Rosemont)
9:00 pm	Grand Opening and Awards (Rosemont)

Saturday, August 20, 1983

7:30 am	Registration Booth Opens
8:00 am	Vendor Exhibit Area Opens
8:00 am	Morning Discussion Groups Begin
9:00 am	NUA Lounge Beginners Seminar
10:30 am	Late Morning Discussion Groups Begin
12:45 pm	Afternoon Discussion Groups Begin
2:00 pm	NUA Lounge Beginners Seminar
3:15 pm	Late Afternoon Discussion Groups Begin
5:00 pm	Vendor Exhibit Area Closed
5:30 pm	Registration Booth Closed
6:30 pm	Dinner Warm-up Open (United A&B)
7:30 pm	Dinner, Keynote Address and Prizes (Rosemont)

Sunday, August 21, 1983

7:30 am	Registration Booth Opens
8:00 am	Vendor Exhibit Area Opens
8:00 am	Early Discussion Groups Begin
10:30 am	Late Discussion Groups Begin
1:00 pm	Registration Booth Closes
2:00 pm	HUG General Meeting Begins
3:30 pm	Closing Remarks
4:00 pm	Close of Second National HUG Conference

National HUG Conference
Discussion Group Schedule
August 20, 1983

Morning Session (Early)

Time	Room	Subject	Speaker
8:00 am	(C)	Communications/Modems	Dale Lamm
8:15 am	(D)	Z100/S100 Interfacing	Jim Buszkiewicz
8:30 am	(B)	ZDOS 2.0	Brian Barnes
8:45 am	(A)	Robotics/Educational Products	Ron Johnson
9:00 am	(NUA)	Introduction to Computing	Gerry Kabelman

Morning Session (Late)

10:30 am	(D)	CP/M Panel Discussion	(Panel)
10:45 am	(B)	H8/H89 Hardware	Bruce Denton
11:00 am	(A)	Z100 Interlace Mode	Dave Perkins
11:15 am	(C)	Software - Heath/Zenith	Bill Zurney

Afternoon Session (Early)

12:45 pm	(C)	Communications/Modems	Dale Lamm
1:00 pm	(D)	Z100/S100 Interfacing	Jim Buszkiewicz
1:15 pm	(B)	ZDOS 2.0	Brian Barnes
1:30 pm	(A)	Robotics/Educational Products	Ron Johnson
2:00 pm	(NUA)	Introduction to Computing	Gerry Kabelman

Afternoon Session (Late)

3:15 pm	(D)	CP/M Panel Disscussion	(Panel)
3:30 pm	(B)	H8/H89 Hardware	Bruce Denton
3:45 pm	(A)	Z100 Interlace Mode	Dave Perkins
4:00 pm	(C)	Software - Heath/Zenith	Bill Zurney

August 21, 1983

Morning Session (Early)

8:00 am	(C)	Communications/Modems	Dale Lamm
8:15 am	(D)	Z100/S100 Interfacing	Jim Buszkiewicz
8:30 am	(B)	ZDOS 2.0	Brian Barnes
8:45 am	(A)	Robotics/Educational Products	Ron Johnson
9:00 am	(NUA)	Academic Computer Uses	To Be Announced

Morning Session (Late)

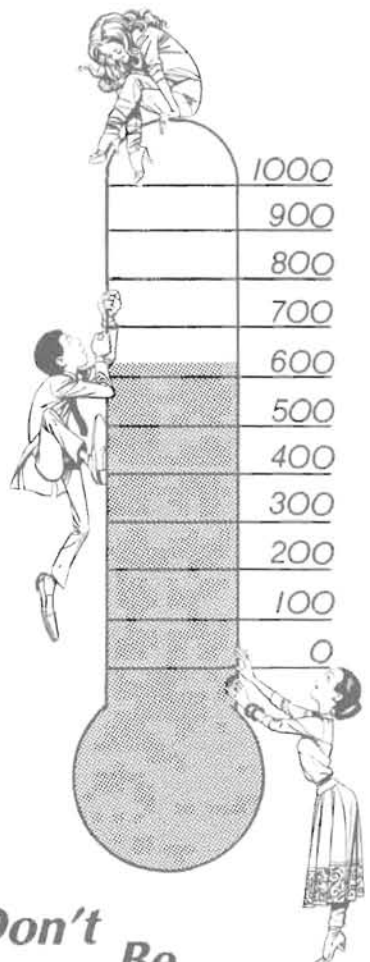
10:30 am	(D)	CP/M Panel Discussion	(Panel)
10:45 am	(B)	H8/H89 Hardware	Bruce Denton
11:00 am	(A)	Z100 Interlace Mode	Dave Perkins
11:15 am	(C)	Software - Heath/Zenith	Bill Zurney

Afternoon Session

2:00 pm	(Rosemont)	General Meeting	Local Club Presentation
2:45 pm	(Rosemont)	General Meeting	Public Domain Software
3:30 pm	(Rosemont)	General Meeting	Closing Remarks

** Speakers for the General Meeting will be announced on the final schedule available at the Registration Booth.

The 1983 HUG National Conference Is Filling Up



Don't Be The One Left Out, Send Your Registration Form In Now!!

Heath Users' Group

NATIONAL HUG CONFERENCE II

Official Conference Registration Form

O'Hare Hyatt Regency Hotel, Chicago, Illinois

August 19, 20 and 21

Name(s) _____

Address _____

Company _____

City _____ State _____ Zip _____

Enclosed is $20.00 per individual to attend The Second National HUG Conference to be held the weekend of August 19, 20 and 21 1983. Please send ticket(s) and information regarding hotel reservations.

AMOUNT ENCLOSED _____ NUMBER ATTENDING _____

For our information:

Which Heath/Zenith computer do you now operate? _____

Are you a Non-User-Attendee? ☐ YES ☐ NO

Are you a Heath/Zenith related vendor? ☐ YES ☐ NO

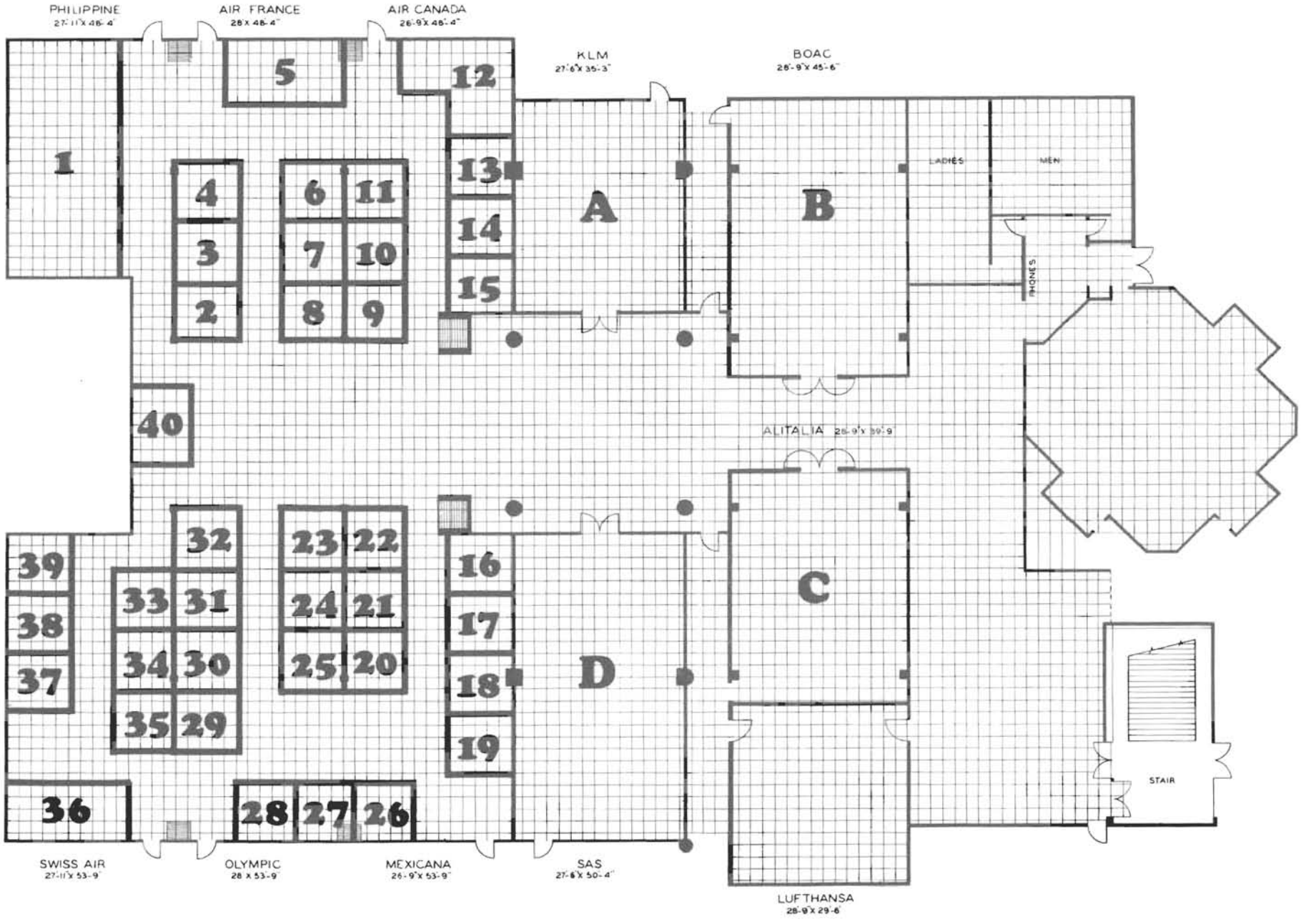
For your information:

Space limitations for the dinner to be held Saturday August 20, 1983, will restrict the number of attendees for that dinner to 1000. Therefore, it is important that you register as soon as possible. Visitor tickets for those of you simply attending and not planning to stay for the dinner and prize drawings will be available at the registration booth for $10.00. Send your registration form or a suitable copy to:

Heath Users' Group
Attention: National HUG Conference Registration
Hilltop Road
Saint Joseph, Michigan 49085

Special Note to Vendors:

Vendor Information Packages will be made available to Heath/Zenith Related Vendors who are planning to exhibit their products while at the conference. Three times more space is available this year for the purpose of showing those products of interest to owners of Heath/Zenith computer products.



Second National HUG Conference

Vendor/Booth Locations

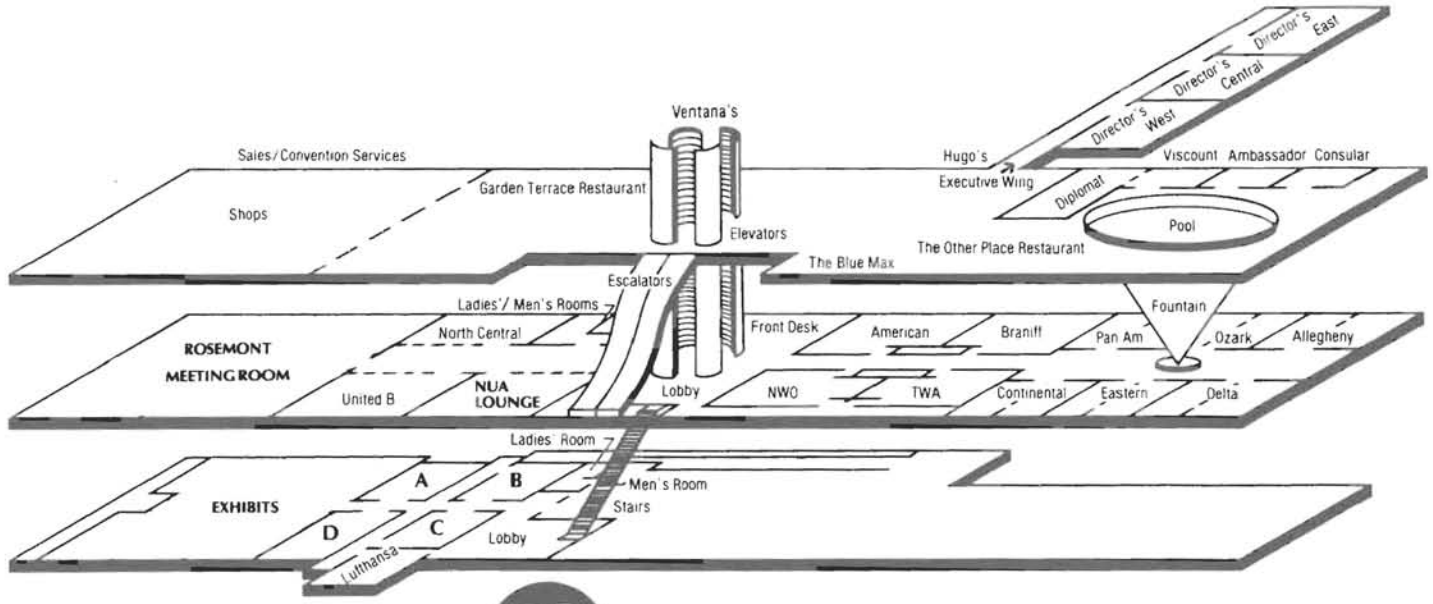
Booth	Vendor
1	Heathkit Electronic Center
2	Magnolia Microsystems
3	Computer Systems Inc.
4	ARTRA
5	dotted i softStuf and Generic Software
6	Valley Data Sciences
7	Hoyle & Hoyle
8	Sunflower Software
9	D-G Electronic Development
10	The Software Toolworks
11	Cleveland Codonics
12	M P I and Studio Computers
13	Ashton-Tate
14	Interactive Microsystems
15	Sextant Publishing
16	Ultimate Computer Systems
17	Greenbrier Marketing International
18	Access
19	Extended Technology Systems
20	C D R

Booth	Vendor
21	Cherry Engineering
22	Kres Engineering
23	Software Wizardry
24	Codewright
25	Quikdata, H-Scoop, Ultimeth
26	Sigma-Soft & Systems
27	Tex-Matics
28	Steve Robbins
29	Hampton Business Machines
30	Newline Software
31	New Orleans General Data Services
32	Apogee Software
33	Jay Gold Software
34	Technical MicroSystems
35	Powerline Systems
36	Sjulstad Engineering and Micro World Publishing
37	Zeducomp
38	Northwest Digital Systems
39	Pat Dengle
40	Heath Users' Group

RESTAURANT LEVEL
Restaurants & Shops

MAIN LOBBY LEVEL
Registration Booth
NUA Lounge
Rosemont Meeting Room

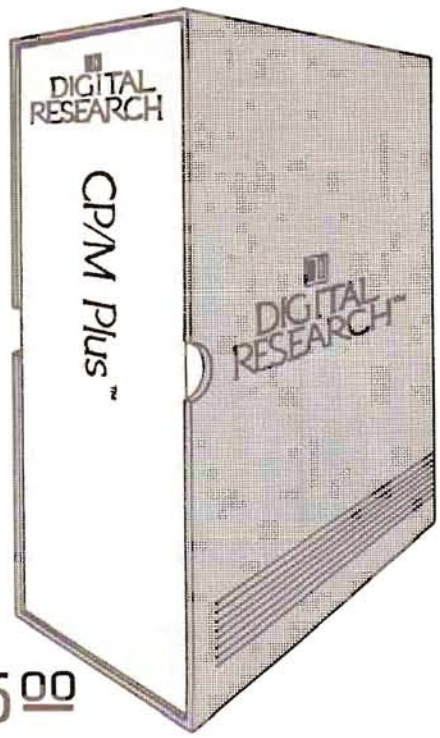
LOWER LEVEL
Vendor Exhibits
Meeting Rooms A, B, C, D



HYATT REGENCY O'HARE

MAGNOLIA MICROSYSTEMS

CP/M Plus™ Released!



Under development since late last year, CP/M-Plus[tm] support is finally available for Z89 and Z90 computers with our 128K RAM board and either Zenith's Z89-37 (Z90) or our own 77316 Double Density disk controller.

With the banked RAM available on our 128K board, disk performance is dramatically enhanced through the use of Hashed Directory tables, Directory Buffers, and LRU Data Buffers.

New utilities and features include a 'Help' command; optional Password protection and Time and Date Stamping of files; Console Redirection to or from disk files; and many others.

Digital Research's list price for CP/M-Plus is $350, but as an introductory special, it will be included AT NO EXTRA CHARGE with our 77318 128K RAM board if purchased before September 1, 1983!

Our implementation of CP/M-Plus REQUIRES the use of our 128K RAM board. We have no plans to implement the non-banked memory version because most of the advantages of CP/M-Plus are not available (or practical) in the non-banked version.

We are including the SOURCE code for our BIOS, together with Digital Research's MAC, RMAC, LINK, and SID software development tools, so you may make whatever modifications necessary for your specific application.

When ordering, be sure to specify which double density controller and size drive you boot from: Zenith's Z89-37 (Z90) or Magnolia's 77316 (5-inch or 8-inch).

Customers who purchased the 128K RAM board after our announcment of CP/M-Plus at CP/M-83 in San Francisco (who provide satisfactory proof-of-purchase) can obtain CP/M-Plus for the nominal $50 cost of the CP/M-Plus documentation (plus shipping and handling).

Earlier purchasers of the 128K RAM board who are also registered owners of the Magnolia Microsystems release of CP/M 2 (who provide satisfactory proof-of-purchase) can update to CP/M-Plus for $100 (plus shipping and handling).

$595.00

128K RAM Board with CP/M-Plus[tm]

Double Density Controller reduced $100

now only $495.

Save $100 on the most versatile controller available for the Z89 or Z90! Supports four 5-inch and four 8-inch single or double-sided Shugart compatible drives. Read and Write over 27 different (including all current Zenith Z89/Z90) media formats.

Everything is the same as before except the price! Includes a copy of CP/M[tm] 2.2, cables for both 5- and 8-inch disk drives, and a new Monitor (Boot) EPROM.

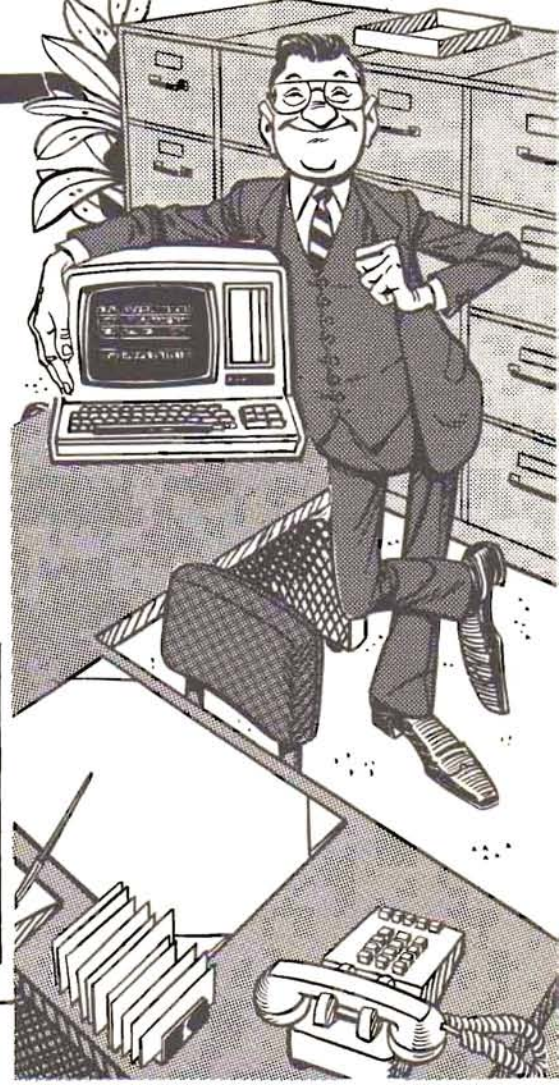
Subsystems reduced also -- take $100 off of all the double-density floppy disk subsystems shown in our price list dated 3/11/83. For example, our popular Dual 8-inch Double-Sided Subsystem was $2695, now only $2595!

MAGNOLIA MICROSYSTEMS, INC.
2264 - 15th Ave West • Seattle, WA 98119
(206) 285-7266 • (800) 426-2841

Keyword-based Filing and Retrieval:

A Specialized DBMS for the H/Z-89

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



About the Author:

Jim Tysinger is a manager of data processing systems at Westinghouse Electronic Corporation. He has been involved in business data processing for fifteen years, mostly in the hardware planning and operating system software areas. Prior to these responsibilities in large computer systems, he developed digital process control programs in assembly language. Personal interests include photography and electronic equipment. He began using his H-89 in 1980 for record keeping tasks at home, resulting in the development of the ideas and methods presented in this article.

Abstract

General data base management systems provide a framework and structure for the orderly collection, storage, manipulation, and display of operational data. These general systems are frequently complex, both in the initial data base setup and in the use of the system to generate reports. Specialized data base management software is available for specific functions such as keeping a business inventory or maintaining mailing lists, or for the somewhat less specific function of filing and retrieval. Although functionally-specialized systems are more restricted in application than general systems, well-designed specialized systems are easier to use, especially for those with little or no data processing experience. In addition to the issue of complexity, the performance-related elements of processing speed and data storage capacity can also limit the practical usefulness of any of these systems in the personal computer environment. This article addresses the specialized data base management function of filing and retrieval by means of keyword association, and offers an approach to dealing with the limitations of processing speed and data storage capacity in desktop machine configurations.

Background

It is useful to make a distinction between "data" and "information". Data can be defined as a collection of facts, and can consist of text material, character strings, numeric values, or all of these in combination. To use a general DBMS, the user must begin by developing the design of the data base which will contain his collection of facts. A definition statement must be associated with the data to describe the various data elements. To illustrate, the table in Figure 1 is provided to represent a simple data base. The values in the table are the data elements. Each row represents one record in the data base. Other names for a data base record are data base entry or data base item. The data definition statement is represented by the column headings. Note that in our example, the column headed "License Number" contains an element of data which is unique for each record, and can be used to distinguish a specific record from all others. This is the "primary key" in our simple data base. There may be more than one column containing a data element which is unique for each record. This means that there may be several candidates for the primary key. One of these candidates is chosen as the primary key by the user, who is usually his own data base designer in the personal computer environment. Data definition, the relationships of data in records, and the existence of the primary key make it possible to apply a process to our data base and display the results in an orderly manner.

Figure 1.

Vehicles of the XYZ Auto Rental Agency

License Number	Vehicle Type	Year Purchased	Color	12/31/82 Mileage	Condition
F06298	Car	1980	Green	25700	Good
F64281	Car	1981	Green	15400	Good
G90703	Truck	1981	Red	15700	Good
R63704	Van	1979	Yellow	38600	Fair
G83421	Truck	1980	White	20400	Fair
F04711	Car	1979	Blue	51900	Fair
F04621	Car	1979	Blue	66070	Poor

Information can be defined as knowledge derived through the processing of data. This processing is usually preceded by the framing of an inquiry to be applied against the data. If the data is organized appropriately, the inquiry can be processed quickly and completely, providing the desired information. In the case of our simple data base, we could ask, "How many vehicles have more than 50,000 miles of usage?" The identification of the two cars having this characteristic provides the desired information. Unique identification is guaranteed by the primary key. In addition to answering an inquiry, many data base systems provide the ability to format and display the entire data base, ordered by all or part of the record content.

Such an ordered list may display desired information. In summary, a fundamental feature of data base management systems is to provide the structure for the derivation of information from data.

A Filing and Retrieval DBMS

So far, some characteristics of general data base management systems have been discussed. No description of the processes of data definition or data collection was given, nor was there any discussion of the language of the inquiry or of the structure of the data base. In the case of general systems, these topics could be the subjects of other articles or in fact, books. If some or all of the above structures and processes are pre-defined and incorporated into the data base manager, then that system can be considered to be a specialized type of data base manager. Filing and retrieval by means of keyword association is one such type of specialized data base management. The table in Figure 2 represents a data base which we can use to illustrate filing and retrieval by keywords. The data is taken from the catalog of REMark articles in Issue 24. The structure is similar to the previous example, but in this specialized case, much of the data definition can be incorporated into the software of the filing and retrieval system. Some flexibility can be retained if at least one field is definable by the user. The primary key is supplied by the system as a unique sequence number generated as the item is stored. It is useful to implement variable-length fields for the keywords and the abstract, so that neither the number of keywords associated with an item nor the length of the text contained in the abstract is restricted by an arbitrary limitation. For the purpose of the illustration, the content of the abstract is omitted, but can be assumed to be text material of user-desired length in each item. The item can contain all of the filed data or as in our illustration, the item can refer to the full text of material which is not contained in the data base.

Retrieval of items from the data base is accomplished by means of keyword association. An inquiry into the data base is set up by specifying a search argument containing keywords, which the system then compares with those associated with each item at the time it was stored. The items meeting the search criteria are selected from the data base for display. For example, the specification of GAMES as the search argument would cause items 2, 3, and 5 to be selected. Depending upon the features of the system, the search argument may be able to accommodate logical operators (AND, OR, NOT), and may be able to use a date as part of the search criteria. If we used an inquiry specifying that both GAMES and ASSEMBLY be associated with items to be retrieved, then only item 5 would be selected.

The keyword association mechanism works well as long as the same keywords occur to the user at retrieval time as those which were used at the time the items were stored. If the use of keywords is inconsistent, the retrieval process will not produce the intended results. Some of the data may be unable to contribute to the development of the information if items are omitted in the selection process as a result of keyword inconsistency. To avoid this problem, some thought must be given to the keywords used when the items are stored. The system can help by providing the user with the ability to request a list of the keywords used in the data base. The use of a scheme to select productive keywords and avoid those which are inconsistent, redundant, or useless, is called a keyword strategy. This strategy may not be completely clear in the user's mind at the time he decides to build a data base. The variable-length keyword field makes it possible to associate many keywords with each item and then pick up the trends from the total keyword listing as the data base grows. The user's skill at developing keyword strategies usually improves rapidly as a result of experimentation.

Figure 2.
A Catalog of REMark Articles

Item #	Date	Subject	Issue/Page	Keywords	Abstract
1	1/1/78	A Numismatic Inventory	1/21	BASIC, COINS, INVENTORY	(comments)
2	1/1/78	Mastermind	1/23	GAMES, BASIC, MASTERMIND	(comments)
3	1/1/78	Mini-Nim	1/24	GAMES, BASIC, MININIM	(comments)
4	1/1/78	Sorting Strings	1/26	SORT, ORDER, BASIC	(comments)
5	10/1/79	The Eight Queens Problem	8/27	ASSEMBLY, GAMES	(comments)

Filing and Retrieval Applications

The opportunities for application of a filing and retrieval system are restricted only by the imagination of the user. Applications range from keeping lists of names (customers, club members, subscribers, etc.) to building an on-line reference library or preparing a bibliography for the writing of a book. For example, a keyword-association system is effective in building a data base of problem definitions based on symptoms described by keywords. Such a data base can aid in the diagnosis of problem situations where a logical combination of symptoms leads to a set of potential or probable specific causes. Other useful applications include an on-line catalog of magazine articles, books, or other prepared material pertaining to a field of technology or a study of some physical science. The amount of data necessary to do an adequate job in some of these areas may overwhelm the microprocessor environment, but it is sometimes possible to use small machines where the scope of data can be limited. The process of building a data base for one of the purposes mentioned can actually help an individual develop a sound, analytical approach to the organization of the pertinent data. The computer system can become an extension of the user's ability to make connections between elements of data that may otherwise go undetected.

Until recent years, only simple tools were available to aid in these processes. Card index systems or filing procedures based on some ordering of the elements of stored data are examples of such tools. The advent of personal computers at affordable prices and the development of data base management software for them has resulted in the availability of more sophisticated tools for storing, organizing, and displaying selected data. As electronic calculators have evolved

into a popular tool for accurately and rapidly performing tedious processes of mathematics, a keyword-association data base management system can become an effective tool in organizing data and developing information.

Data Collection and Distribution

Unless the internal structure of the data base is simple enough to permit data addition, modification, and deletion directly by means of an editor, these functions must be addressed by the data base management system. If the data base manager uses any internal indexing techniques to facilitate location of stored data, the provision of a data collection capability is a requirement. This is one area where "user friendliness" comes into play. The system can provide a prompt for each field of data, along with some indicator of the field length to make data entry an easier task. A facility for immediate modification of entered data is a helpful addition. Error correction is not usually a problem in interactive data entry mode. If the user violates a format convention monitored by the system, an error indication can be given, and the user can correct the problem before proceeding. On the output side, many systems offer the user a choice of displaying results on the screen or printing them on a hardcopy device. These input and output schemes satisfy the requirements of the user in what may be called a "closed" environment. The term "closed" can be applied to an application which requires input only from the keyboard and delivery of output only to display devices such as a CRT or printer.

A means of transmitting data in machine-readable form either to or from a data base management system is an additional capability which may save the user many hours at the keyboard. A DBMS having these capabilities provides a definition of interface specifications to the input process and from the output process. A user can prepare a file according to the input process specifications and feed that file into the data base manager instead of keying the data directly into the DBMS. If the user has already collected data in text files, there is a possibility that most of the effort of preparing those files can be salvaged. A program written in BASIC or some other appropriate language may be able to re-format the data into a form acceptable as input to the DBMS. On the output side, the user may wish to use output from the DBMS as input to a program he has prepared. In this case, such a program may be written to accept data in the form supplied by the DBMS, or another program may be used to re-format the data. The operations described here may be termed "batch mode". Operations where the user interacts directly with the software in each transaction can appropriately be called "interactive mode".

Some provision must be made for the correction of format errors which may occur during operation in batch mode. One way the DBMS could handle the situation would be to produce a transaction report containing error indications for format errors. A second method would be to switch to interactive mode for the correction and then resume operation in batch mode. Either method can provide enough flexibility to give the user a reasonable way to handle errors when operating in batch mode.

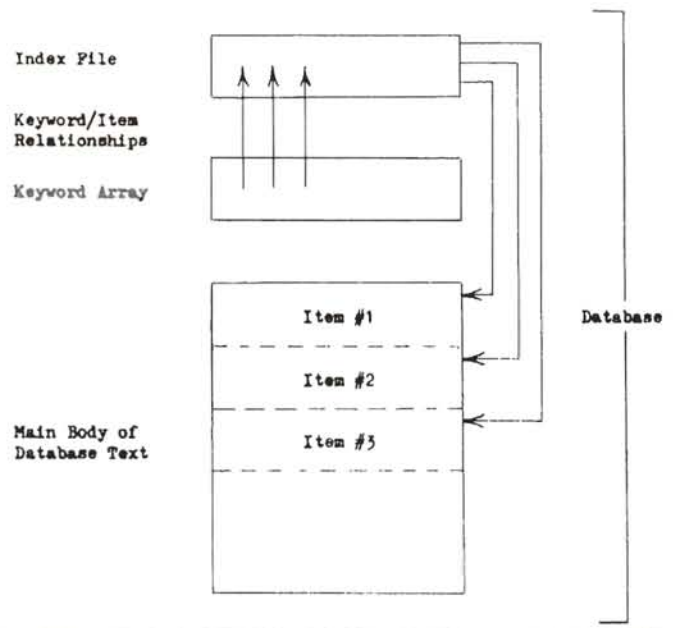
Processing Speed

One of the strengths of microcomputers is the speed with which strings of characters may be manipulated in main memory. Logical operations, character comparisons, and simple arithmetic can be performed at the hardware level with single machine instructions. These functions do not require whole routines or sets of multiple operations to complete, as do routines for square roots, trigonometric functions, and even multiplication and division on the 8080 or Z80-based machines. Selective retrieval of data based on logical functions, character string comparisons, and other simple operations can therefore be accomplished very quickly by 8080 or Z80 machines. The bit manipulation capability of these machines allows storage of the results of comparisons in bit arrays, permitting not only high speed access but also very dense packing of these results. Hundreds of character comparisons can be made, the results stored in bit arrays, and logical decisions performed on those results in one or two seconds of elapsed time. All of this can lead to fast selection time as long as the data to be used in the selection process is resident in main memory, frequently referred to as RAM (Random Access Memory). When selection data must be streamed from the disk, the relatively slow access time and data rate of this device significantly affects retrieval time.

One search technique which avoids disk accesses during the selection process operates on a RAM-resident array instead of the actual disk-resident data base. In this technique, the searchable portions of stored data (in our case, the keywords) are extracted from the data base and are encoded by the DBMS into a densely-packed array which can be contained in RAM. The keywords are compressed and indexed to avoid an unreasonable limit on the volume of searchable data. A system using this technique will work best, however, in computers having the fully-expanded RAM configuration. The extraction process can occur at a time when it does not cause lengthy and annoying delays. As an example, ordering and indexing of keywords can be accomplished as the data is entered into the data base, "stacking the deck" so to speak, for the search technique. The resulting data base consists of several physical files as illustrated in Figure 3. These files are used as a coordinated set by the DBMS. The index and keyword array files are usually relatively small compared to the main file. To achieve the full speed bene-

Figure 3.

A Keyword DBMS File Layout



fit, they must be RAM-resident during the use of the data base to which they belong. In addition to providing greater selection speed, these compact files provide the key to addressing large data bases. Large data bases are discussed in a later section.

Although indexing and keyword extraction offer many advantages, there are some disadvantages. Activity in the data base becomes dependent upon pointers and sector keys in the index files. Updating the data is not a straightforward task within the DBMS software, and recovery of the data base from a damaged file is complicated. The problem of recovery will be discussed later. To address the update problem, a data base unload/reload feature becomes a requirement instead of simply an extra capability. The term "unload" means that data contained in the data base is moved from the internal DBMS structure to an ordinary sequential file to permit modification by software using standard file manipulation techniques. When unloaded, a data base entry can be modified easily by an editor and then reloaded by the DBMS. In summary, indexing and extraction techniques provide a significant selection speed advantage, although adding the unload/reload operations to the task of changing a data base entry.

Large Data Bases

Personal computers are well suited for the data base management application if the limited size of the mass storage media (floppy disk) is not a problem, or if the system can effectively deal with this constraint. Space conservation is important in data bases used in the desktop computer environment. The use of special terminator characters to define the end of fields and the end of records in the internal data base structure prevents the loss of space resulting from the implementation of fixed-length or boundary conventions. Although some systems require each record to begin at a sector boundary, this programming convention is wasteful of space and is not dictated by hardware or storage media constraints. Even with careful conservation of disk space, however, many practical DBMS applications require a very large space for the data base.

File access methods in use on microprocessors today do not usually permit a single file to span several diskettes. Such a file would be very clumsy to manipulate, even if the access methods allowed it. It is not a bad practical approach to divide a large set of data into manageable pieces, but yet consider them as one common collection of data. Each small part is a physical data base, and the whole logical data base is made up of all of these physical parts. This leads to techniques for allowing a large logical data base to consist of a number of physical data bases which are considered by the DBMS to be connected during some of its functions. Because each physical data base can be updated or modified individually, the obvious function which must allow for the logical connection of physical files is the search. There are probably many ways to accomplish this, only one of which will be used to illustrate the point. If the data base files use specific three-character file extensions, then the DBMS can select its data base files from the contents of a diskette. In this scheme, the user would place physical data bases to be treated as part of a large logical data base together on one or more diskettes. The large logical data base would consist of several diskettes containing only physical data bases that belong to the same set. As long as the DBMS software can examine the contents of a diskette volume (DIRECT.SYS), discriminate between file types by means of the file name extension and provide a dismount/mount feature, all physical data bases on many separate volumes could be searched with one specified search argument.

When the number of volumes to be searched becomes large, dismounting and mounting diskettes becomes tiresome. The indexing technique discussed earlier provides a means of dealing with this problem. Usually, the index file and keyword extract files are small subsets of the data base. They are also the only files required for the search process. If only these index and extraction files for each physical data base are copied to a common diskette, the searchable portions of many physical data bases can be accommodated by one diskette. This approach is illustrated in Figure 4. A very large data base can be searched with a reduced number of dismounts and mounts using this method. It is true that any change in one of these physical data bases will necessitate a re-copy of its index and extraction files to the search diskette, but many applications requiring extremely large data bases do not require constant updating. If the user is willing to perform a few file-copying chores and keep track of his data base activity, then this multiple data base search technique may offer a solution to the problem of dealing with large data bases.

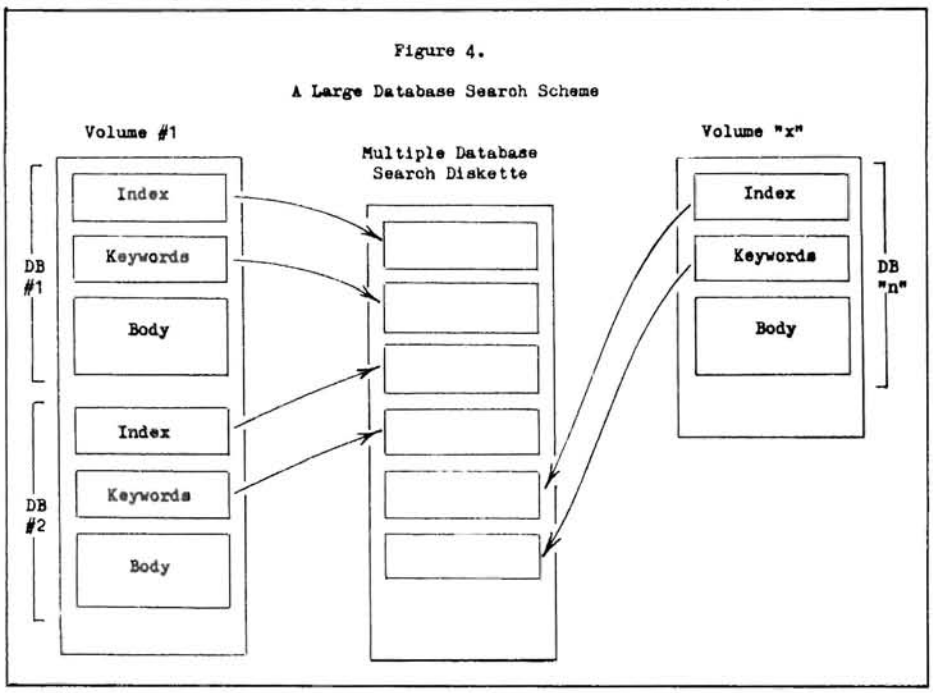


Figure 4.

A Large Database Search Scheme

Backup and Recovery

Few problems can be as discouraging as the loss of a data base containing the product of many hours work. Such problems can occur because of physical damage to the storage media, a hardware malfunction, a procedural error in using the computer, or an error in the software system. Data base systems using indexing techniques and data compression methods are more vulnerable to hardware or procedural errors (and program bugs) than systems using ordinary sequential files. Although well-designed and thoroughly-tested systems can go a long way toward preventing the loss of data, no system can be absolutely safe in this regard. Every user will eventually encounter that situation where there is a fervent effort to "put things back the way they were" before the catastrophe occurred, rather than start over from scratch. A data base unload capability may be of some value in the case where the internal linkages have been damaged. If the unload is completely successful

or recovers a large part of the data base, the work involved in rebuilding it is appropriately reduced. For this reason, the unload function should contain some diagnostic capability. If the problem is a physically damaged diskette or a deleted file, there is some software around for possibly recovering the files, although luck plays a substantial role in the success of this recovery process.

The best procedure by far is one which operates primarily before the fact rather than after the fact. The user's exposure to loss of data can be significantly reduced by performing frequent backups of data bases. Although this procedure requires additional diskettes and the discipline to copy files and keep a log of activity, it is the most simple and straightforward way to maintain an escape route for use in those unfortunate times. The timing of the backup copies is determined by the frequency of the update sessions and by the amount of data entered in each session since the last backup. Common sense is a good regulator of the amount of vulnerability the user can tolerate. This may not be a very sophisticated approach, but the simple procedures work best in some cases.

An Implementation

It is frequently easier to discuss ideas than it is to develop a software system which implements them. There is, however, a filing and retrieval system which is designed to incorporate the ideas discussed in this article. AUTOFILE Version 3.0 (HUG Part Number 885-1110) was written to provide a filing and retrieval capability in the HDOS environment. The data base structure supported by AUTOFILE will accommodate the example described by Figure 2 and similar examples discussed under Filing and Retrieval Applications. AUTOFILE is specifically written for the H/Z-89, as it uses the full Z80 capability to optimize speed and efficiency of operation. While it will function in a single disk drive configuration, two drives provide much greater flexibility for large data bases. Further operational information on AUTOFILE can be found in the HUG software catalog and in Issue 23 of REMark.

Big News
"Odds and Ends"
Sale from Heath!

With the approach of the Second National HUG Conference, Heath Company is announcing a "last chance" sale on miscellaneous stock items. These items can be ordered from your local Heathkit Electronics Center or mail order should you desire. The following is a list of the items available. Items that are not sold before the National HUG Conference will be available at the Heathkit Electronics Center Booth established within the O'Hare Hyatt Regency during the Conference.

H8 Equipment	Description
H-8	H8 Computer
H-8-2	Parallel Board
H-8-5	Serial Board
H-8-9	PAM-GO ROM
H-8-10	Wire-Wrap Board
H-8-19	Z80/37/67 Key Caps
H-17	Floppy Disk with Supply
HA-8-1	Extender Board
HA-8-3	Color Graphics Board
HA-8-6	Z80 CPU Board
HA-8-8	Orgin 0 Modification
WH-8-41	4-Port Serial Board
WH-8-37	Soft-Sectored Controller Board
WH-8-47	8-inch Interface Board (H/Z-47)
WH-8-51	Molex to "D" Connector
WH-8-64	64K Memory Board

H-11 Equipment	Description
H-11-1	4K Memory Board (Kit)
WH-11A	Wired H-11
WH-11UL	UL Approved H-11 (Wired)
WH-11-5	Serial Board (Wired)
WH-11-51	Molex to "D" Connector
WHA-11-5	Serial Board (Wired)
WHA-11-16	16K Memory Board (Wired)
WHA-11-32	32K Memory Board (Wired)

Information on pricing or further information about the products described above can be obtained by calling the Computer 800# for all calls except those originating in Michigan.

Computer Line 1-800-253-7057
Michigan 1-616-982-3285

To give you an idea of the price reduction, take a look at the following!

WH-11A Wired H-11 Computer
Normally $1995.00 Now $800.00

Win up to $500 Worth of Prizes!!!

Now is your chance to cash in on your robotics programming skill and creativity. Enter the first Microcomputing/Heath Company HERO 1 programming contest and win up to $500 worth of prizes. Microcomputing magazine, in conjunction with the Heath Company, manufacturers of the HERO 1, invites all HERO 1 programmers to submit their best applications to this contest. Entries will be judged in the following categories:

1. Standard HERO 1 with arm.

2. Modified HERO 1, including additional RAM or ROM, as well as any mechanical or electrical modifications.

Prizes will be awarded to the top three entrants in *each* category. Two $500 gift certificates (one from each category) will be awarded. Each first place winner will select the prizes of his choice, worth up to $500, from the latest Heath Company catalog. A $100 gift certificate, good toward any purchase from the Heath catalog, will be awarded to both second place winners. Third place winners from each category will receive a copy of Microcomputing columnist Mark Robillard's new book, "HERO 1 Advanced Programming and Interfacing," plus a one-year paid subscription to Microcomputing magazine.

CONTEST RULES

1. All programs must be submitted both on cassette tape and in hard copy form. A brief, written description of the application must accompany each entry.
2. Entries in the modified category must include a complete description of the alterations performed on the robot.
3. The contest is open to all HERO 1 owners, except employees of Wayne Green Inc. (publisher of Microcomputing), and the Heath Company and and their immediate families.
4. All entries, including programs, become the property of Microcomputing.
5. All entries must be received by Microcomputing by September 1, 1983.

Send submissions to:

Robotics Contest
Microcomputing
80 Pine Street
Peterborough, N.H. 03458

7. Contestants may submit more than one entry in one or both categories. Entries will be judged on originality and technical feasibility. The more practical and easily adaptable the application, the better. Winners will be announced in the December 1983 issue of Microcomputing. So rev up your robot, and let's put the Heath's HERO through its paces!

MICROCOMPUTING®

80 PINE STREET PETERBOROUGH, NH 03458

A Z-100 Font Editor

Marc Aagenas
ZDS Software Consultation

We are driven! That is, we in Software Consultation do attempt to respond to the needs of our customers. Often times a customer will call with a question that we can't answer straight away, at least not without some research. Well, from just such a question regarding the use of the ALTCHAR.SYS file on the Z-100 comes this article and program. In learning how the ALTCHAR.SYS file is used, I wrote a program to create and modify this type of file. It is called FONTED.BAS and runs on the Z-100 under ZBASIC.

Introduction

The Z-100 is a study in flexibility. Eight and sixteen bit microprocessors, the S100 bus, monochrome and color graphics, and, a soft keyboard/character-set. This flexibility surrounding the keyboard and display makes it very easy to market the computer in almost any country in the world. With a simple change of key caps and the ALTCHAR.SYS file, the Z-100 contains the proper keys and display characters to make it feel as at home in France or Italy as it does in the U.S.A. But this tractability shouldn't be limited to just supporting Foreign keyboard layouts and character fonts. Instead, it opens up the possibility for useful alternate graphics and interesting keyboard (Dvorak?) layouts. Maybe a game disk with specialized graphics, modified ASCII characters and a remapped keyboard. I will leave the creativity up to you, the program to implement it follows the article.

What is ALTCHAR.SYS ?

What is ALTCHAR.SYS ? A common question at software consultation, with a fairly simple answer. ALTCHAR.SYS is a file which contains the keyboard mapping, character font data, and display mapping changes/additions in information used by Z-DOS and CP/M85 to decode and display characters. That sounds like a lot but it is divided into three distinct parts, each of which are quite simple. The Z-100 uses three tables for input/display. The keyboard map, the font data table, and the display map. At boot time, each table is initialized to its default condition and then ALTCHAR.SYS is read in and its contents are used to make changes in each of the standard tables.

The keyboard map contains 255 entries. Each key is assigned a location in the key map and the default value of a key is equal to its location in the map. An example follows:

```
        (legend  c = control, sh = shifted)
map location  :   0   1   2   3 ... FE  -(these numbers are in
char assigned :   0   1   2   3 ... FE  -hexadecimal and they
key struck    :csh2  cA  cB  cC          come from the Z-100
char generated:  c@  cA  cB  cC          Users Manual p. B.10)
```

From the example, if you struck a control C, the keyboard would generate a 03 and in our default map above, a 03 (the key struck) would be sent to Z-DOS. Now from Appendix B of the Z-100 Users' Manual, we know that 03 is the value assigned to control C and that is what we get. But if we modify this keyboard map, we can get different characters generated by substituting the char assigned value (see examples) with a new value.

```
        (legend  c = control, sh = shifted)
map location  :   0   1   2   3 ... FE
char assigned :   0  21  2A   3 ... FE
key struck    :csh2  cA  cB  cC
char generated:  c@   !   *  cC
```

In this modified keyboard map, if we struck a control B, we would get an exclaimation point sent because the map has that character assigned to the control B map location. In effect, we have remapped the control B key code.

These changes are what you find in ALTCHAR.SYS for the keyboard map. This part of the file is terminated by 2 FF's (hexadecimal). If you examined the GRAPHICS.CHR file on your disk, you would find that the first two bytes in it are FF's, why, because there are no changes in the keyboard map. It contains only the graphics data font and the display map.

The character font data section of the file is structured differently, for each character in the file there is a descriptor or index byte and nine data bytes (the actual displayed character). This portion of the file ends with an FF byte as the descriptor. The data for each character looks like this:

```
Descriptor:     Data bytes:
                                  76543210
                                 +--------+
   21H            00H        1 |        |
                  1CH        2 |   ***  |
                  22H        3 |  *   *  |
                  22H        4 |  *   *  |
                  3EH        5 |  *****  |
                  22H        6 |  *   *  |
                  22H        7 |  *   *  |
                  22H        8 |  *   *  |
                  00H        9 |        |
                                 +--------+
```

Note that the value of the descriptor is 21 Hex, not 41 Hex as you would expect for the letter "A". All the descriptors for non-graphics characters are the ASCII value of the character minus 32 (20 Hex). This allows more efficient use of memory since the first 32 ASCII codes are non-printable. The graphics characters start where the lower case characters would normally be but the descriptor value is 2 higher than the ASCII lower case equivalent. See the example below:

```
Descriptor:     Data bytes:
                                  76543210
                                 +--------+
   60H            00H        1 |        |
                  00H        2 |        |
                  1CH        3 |   XXX  |
                  3EH        4 |  XXXXX |
                  3EH        5 |  XXXXX |
                  3EH        6 |  XXXXX |
                  1CH        7 |   XXX  |
                  00H        8 |        |
                  00H        9 |        |
                                 +--------+
```

The ASCII character associated with the graphics bullet or ball is the "↑" which is 5E Hex or 2 less than the 60 Hex of the descriptor. This is done to allow the full ASCII displayable character set enough room in the table since the last character is the "⌐" which has a value of 7E Hex (5E hex as a descriptor in the font table, remember to subtract 32 (20 Hex)). The font data table is terminated by an FF Hex in the descriptor position. This may all seem a bit confusing, but not to worry, the descriptor conversion is handled by the FONTED program and all you need to do is select which character, graphics or otherwise, you wish to work with.

The last section of the ALTCHAR.SYS file is the display map. The display map is where Z-DOS goes to decide what character to display. The map defaults to displaying the character passed to it by the keyboard map, but with modification of the map, you could display the letter Z whenever the key map passed the letter A. I won't go into any further detail on this, since FONTED contains no provisions for modifying the display map, it merely creates a default version.

What is FONTED.BAS?

FONTED.BAS is a program written in ZBASIC that can create or modify ALTCHAR.SYS type files. FONTED.BAS works with the keyboard map and font data tables but does not work with the display map! The program is menu driven and allows selection of which map or table you wish to work with.

FONTED.BAS Cautions!

FONTED.BAS is reasonably well debugged, but there are some Caveats. First, the key map section of the program will allow approximately 60 key changes to be displayed, more than this can be made, but you won't see them all at once. Second, the font data table display cannot exceed 72 characters, there isn't enough room on the screen for anymore characters. These two limitations are not that drastic, but they are limitations that you should be aware of in case you intend to modify the whole key map or font data table. One final reminder, this program does not modify the display map, it merely creates a one-for-one map for the graphics portion. The program could be modified to allow display map changes, a section of code similar to the keyboard map routine would have to be added. I will leave that up to you.

Using FONTED.BAS

Once you have entered the program and feel confident about the accuracy of it, then it's time to give it a try. Start with a disk containing ZBASIC, FONTED.BAS, and all the alternate character files from Z-DOS #1 disk. Run the program and select GRAPHICS.CHR as your working file. When you get to the main menu, select the KEY MAP option and you will see that there are no numbers displayed. This is correct, the file GRAPHICS.CHR is the standard ALTCHAR.SYS file on your disk and there are no changes made in the key map. Exit from this section by typing FF and then enter the FONT option. What you will see displayed are characters in the current ALTCHAR.SYS file and their modified equivalents found in the file you are working with. The graphics characters are an exception to this rule, they are represented by their lower case equivalents.

To examine or add a character to this table, simply type in the character (in proper case) and specify whether or not it is in the graphics section of the table. The program will search the file for the character and if it is found, put an enlarged font matrix with the character in it on the right side of the screen. If the character is not found, then it will assume you want to add it to the file and put a blank matrix on the screen for your use. Using the cursor keys and the function keys, you can turn on or off any of the pixels that make up a character or object. By following the prompts in the 25th line, you can save the character you have modified or created or throw it away. Experiment with this part of the program, change one of the graphics characters, discover what new and interesting characters you can create.

When you are done and back at the main menu, you can optionally save the file back onto disk with any changes you have made. Now to implement this file, you should copy it to an operating system disk and rename it ALTCHAR.SYS.

Experiment, make a practice system disk, modify the ALTCHAR.SYS file on it, and then reboot that disk. Any changes you made will now show up on the keyboard and screen as you use it. Another idea is to try looking at the file FRENCH.CHR. It has many keys remapped and some characters changed in the font table for the special accent markings the French use. If you are daring, try installing this file as ALTCHAR.SYS on your practice disk. Many of the keys are remapped and just trying to use simple commands becomes interesting.

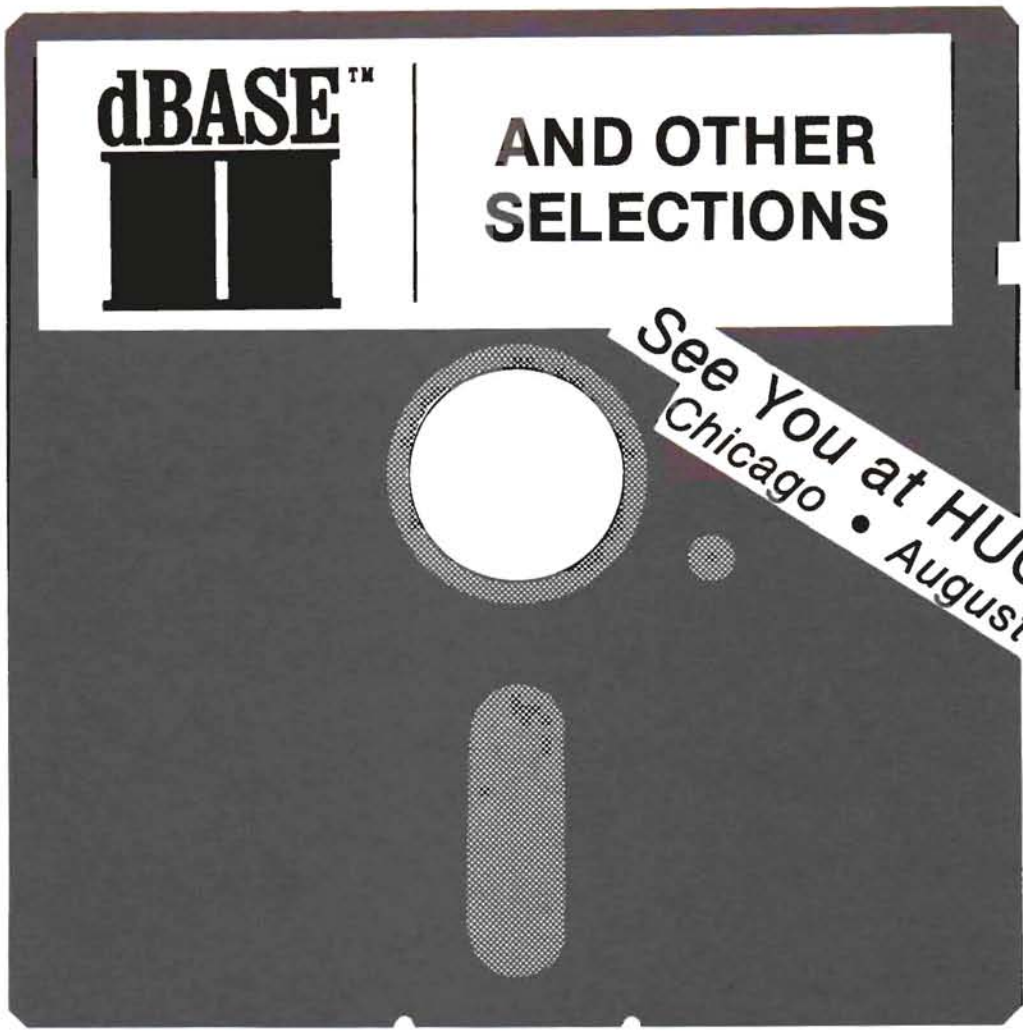
In closing, I hope that this program spawns other programs that use special character sets, graphics, etc. We need a good Chess game, this program can create the men. I also encourage any modifications to this program that you might make. However, because of my position in ZDS Software Consultation, I cannot offer any assistance regarding this program during my normal working hours. Please direct any correspondence to me through HUG. Thank you.

```
10  '            initialization
20 ON ERROR GOTO 2790
30 DIM Z%(74),Z1%(74),Z2%(8),Z3!(433)
40 DIM ARRAY%(2000),KYBD%(255)
50 X=400:Y=35:X1=X+18:Y1=Y+12:CURSORX=X:CURSORY= Y+14
60 '            signon banner
70 CLS:LINE(231,98)-(392,126),2,B
80 LOCATE 12,31:PRINT" FontEd Ver. 1.01 "
90 LOCATE 13,30:PRINT" By Marc O. Aagenas "
100 LOCATE 14,36:PRINT" 5/03/83 "
110 FOR D=0 TO 1000:NEXT D:SCREEN 0,0
120 '           build graphic pixel and font cursor
130 LINE (X,Y)-(X1,Y1),,B:GET(X,Y)-(X1,Y1),Z%
140 PAINT(X+2,Y+2),,7:GET(X,Y)-(X1,Y1),Z1%
150 LINE(X,Y+14)-(X+18,Y+14):GET(X,Y+14)-(X+18,Y+14),Z2%: CLS
160 '           load default values into keyboard array
170 FOR K = 0 TO 255:KYBD%(K)=K:NEXT K
180 '           main program
190 PRINT "Font Editor"
200 PRINT:INPUT "FILE NAME PLEASE -->"; AAA$
210 '           attempt to open file sequentially
220 '           will execute on error if it does not exist
230 OPEN "I",1,AAA$:CLOSE #1
240 '           open and read in file separating into keymap
and font arrays
250 CLS:PRINT "Found file":PRINT"Loading Key Map"
260 OPEN "r",1,AAA$,1
270 FIELD #1,1 AS AA$
280 WHILE ASC(AA$) <> (&HFF)
290           GET #1
300           AA1=ASC(AA$)
310           GET #1
320           KYBD%(AA1)=ASC(AA$)
330 WEND
340 GET #1      'get last FF before font table
350 PRINT "Loading Character Font"
360 FONDX= 0
370 ARRAY%(FONDX)=ASC(AA$)
380 IF ARRAY%(FONDX)=255 THEN GOTO 480
390 WHILE ASC(AA$)<> (&HFF)
```

☞

```
400           FOR X1= 1 TO 9
410           GET #1
420           ARRAY%(FONDX+X1)=ASC(AA$)
430           NEXT X1
440           GET #1
450           FONDX=FONDX+X1
460           ARRAY%(FONDX)=ASC(AA$)
470 WEND
480 CLOSE #1
490 '                 main menu routine
500 KEY 1,"F1":KEY 2,"F2":KEY 3,"F3":KEY 4,"N":KEY 5,"F":KEY
 6,"D":KEY 11,CHR$(5)
510 CLS:LOCATE 10,30:SCREEN 0,1
520 PRINT "Edit Key Map --> <F1>":LOCATE 12,30:PRINT "Edit F
ont --> <F2>"
530 LOCATE 14,30:PRINT"Exit  --> <F3>"
540 LOCATE 16,30:PRINT"Select one ! <  >"
550 LOCATE 16,44:ACT$= INPUT$(2)
560 IF ACT$= "F1" THEN PRINT ACT$:GOSUB 860
570 IF ACT$= "F2" THEN PRINT ACT$:GOSUB 1020
580 IF ACT$= "F3" THEN PRINT ACT$:GOSUB 1050
590 IF ACT$="  " THEN GOTO 510: REM return from a gosub rest
art menu
600 BEEP:GOTO 550
610 '           this routine converts a 2 byte hex string int
o the
620 '           decimal equivalent  (A3) and returns
630 LOCATE 25,29
640 A4=0
650 A$=INPUT$(2)
660 A1=ASC(MID$(A$,1,1))
670 A2=ASC(MID$(A$,2,1))
680 IF A2=13 THEN SWAP A1,A2:A1=48
690 PRINT CHR$(A1)+CHR$(A2);
700 SWAP A1,A4:GOSUB 760
710 IF A4=255 THEN GOTO 630
720 SWAP A1,A4:A1=A1*16:SWAP A2,A4
730 GOSUB 760
740 IF A4=255 THEN GOTO 630
750 SWAP A2,A4:A3=A2+A1:RETURN
760 IF (A4 > 47 AND A4 <58) OR (A4 >64 AND A4 < 71) OR (A4>9
6 AND A4<103)                                          '
            THEN ELSE PRINT CHR$(7):A4=255:RETURN
770 A4 = A4-48
780 IF A4>9 THEN A4=A4-7
790 IF A4> 15 THEN A4=A4-32
800 RETURN
810 '           display all non-default entries in the keybd
array
820 PRINT "Key map changes for ";AAA$;" file";PRINT
830 FOR K = 0 TO 255
840 IF KYBD%(K)<>K THEN PRINT "Key --> "HEX$(K);" = ";HEX$(K
YBD%(K)),
850 NEXT K:RETURN
860 FOR YY= 0 TO 100:NEXT YY:CLS:SCREEN 0,0     ' a short del
ay loop
870 '           start of keybd map modification routine
880 PRINT "The key map you are about to see contains two HEX
ADECIMAL"
890 PRINT "numbers for each key listed. The first number is
the key number,"
900 PRINT "refer to your Z-100 manual Appendix B(p. B-10) fo
r a list of all keys."
910 PRINT "The second number is the swap or new value to be
generated by this key."
920 PRINT "This value represents the key code that will be g
enerated when the key"
930 INPUT  "represented by the 1st number is struck. Please
hit RETURN when
      ready to proceed";ZZZZ
940 CLS:GOSUB 820
950 '           keybd map input handler
960 LOCATE 25,1:PRINT "Input key number in hex -->(FF to exi
t)";:GOSUB 610
970 IF A3 = 255 THEN ACT$="  ":RETURN
980 LOCATE 25,42:PRINT "present value --> ";HEX$(KYBD%(A3))
990 ATEMP = A3
1000 LOCATE 25,65:PRINT "new value --> ";:GOSUB 640
1010 KYBD%(ATEMP)=A3: GOTO 940
1020 SCREEN 0,0:CLS:GOSUB 1360
1030 ACT$="  ":RETURN
1040 '          file save routine
1050 SCREEN 0,0:CLS
1060 PRINT "Save Font and Keyboard Maps as file --> "+AAA$+"
 <Y>";:INPUT;YES$
1070 IF LEN(YES$) = 0 OR YES$ = "Y" OR YES$ = "y" THEN ELSE
CLS:END
1080 LOCATE 3,1:PRINT "Writing File !"
1090 OPEN "r",1,AAA$,1
1100 FIELD #1,1 AS BT$
1110 FOR KDX=0 TO 254
1120 IF KYBD%(KDX)<>KDX THEN GOSUB 1320
1130 NEXT KDX:GOSUB 1320
1140 FOR FONDX= 0 TO 2000 STEP 10
1150 IF ARRAY%(FONDX)=255 THEN 1200
1160 FOR FDX=0 TO 9
1170 LSET BT$=CHR$(ARRAY%(FONDX+FDX)):PUT #1
1180 NEXT FDX:NEXT FONDX
1190 ' add display map to file
1200 FF=255:GOSUB 1310
1210 FOR DISP = 96 TO 128
1220 LSET BT$=CHR$(DISP):PUT #1
1230 LSET BT$=CHR$(DISP):PUT #1
1240 NEXT DISP
1250 FF=255:GOSUB 1310
1260 GOSUB 1310
1270 FF=(&H1A):GOSUB 1310
1280 CLOSE #1
1290 SCREEN 0,1:PRINT:PRINT " Done ! ":SCREEN 0,0
1300 END       ' all done !!!!
1310 LSET BT$=CHR$(FF):PUT #1:RETURN
1320 LSET BT$=CHR$(KDX):PUT #1
1330 LSET BT$=CHR$(KYBD%(KDX)):PUT #1
1340 RETURN
1350 CLS:INPUT "Want instructions ! <N>";INS$
1360 INPUT "Want instructions ! <N>";INS$
1370 IF LEN(INS$)=0 OR INS$="N" OR INS$ ="n" THEN 1520
1380 PRINT "The font editor allows you to create alternate c
haracter fonts for"
1390 PRINT "any of the displayable characters, including the
 H/Z-19 graphics"
1400 PRINT "characters.";PRINT
1410 PRINT "The next screen you see will contain all the mod
ified characters, "
1420 PRINT "their fonts and all the (remaining) graphics cha
racters. If you wish"
1430 PRINT "to change a font or add a font to the file, then
 enter the proper"
1440 PRINT "letter for the character and specify (if needed)
 whether or not the"
1450 PRINT "GRAPHICS equivalent is desired. A enlarged font
will be displayed "
1460 PRINT "and (if found in font file) the current represen
tation of this"
1470 PRINT "character will be displayed. Using the cursor ke
ys you can modify on"
1480 PRINT "on a cell by cell basis any of the cells in this
 character."
1490 PRINT:PRINT "Commmands for saving, deleting and restori
ng can be found on the"
1500 PRINT "bottom of the screen."
1510 INPUT "When ready to proceed hit RETURN ! ";XXXX
1520 CLS:SCREEN 1,1
1525 '          print font display header
1530 RESTORE 1580
1540 FOR LL = 1 TO 5:READ AA$
1550  FOR LL1 = 1 TO 9:LOCATE LL,5*LL1:PRINT AA$;
1560  NEXT LL1
1570 NEXT LL
1580 DATA "CiF","HiO","AiN","RiT","k k"    '
```

```
1585 '          display character position and font table equ
ivalent
1590 SCREEN 0,0:FONDX=0
1600 FOR LL = 7 TO 24 STEP 2
1610           FOR LL1 = 1 TO 9
1620           GOSUB 1850
1630           FONDX=FONDX+XXX
1640 IF ARRAY%(FONDX) =255 THEN ELSE NEXT LL1:NEXT LL
1645 '          25th line display and screen prompts
1650 P1=1:P2=20:P3=40:P4=60
1660 LOCATE 25,P1:SCREEN 0,1
1670 PRINT "Enter Character in proper case (U/L) <F11> = exi
t";:CAR$=INPUT$(1):IF CAR$=CHR$(5) THEN RETURN
1680 LOCATE 25,P3-2:PRINT CAR$+" ";:PRINT "; Graphics Font Y
/N ? ";:GPH$=INPUT$(1):PRINT GPH$
1690 SCREEN 0,0
1695 '          adjust character for proper position in font
table?
1700 IF ASC(CAR$)>93 AND(GPH$="Y" OR GPH$="y") THEN CAR=(ASC
(CAR$)+2) ELSE CAR=(ASC(CAR$)-32)
1710 '                  find character in font table
1720 FONDX = 0
1730 IF ARRAY%(FONDX) = 255 THEN GOSUB 2570:PRINT "NOT IN FO
NT FILE WILL CREATE":GOTO 1770
1740 IF ARRAY%(FONDX) = CAR THEN  ELSE FONDX=FONDX+10:GOTO 1
730
1750 GOSUB 2570       ' clear 25th line
1760 PRINT "FOUND CHARACTER WILL DISPLAY"
1770 LINE1=99:DOT1=600
1780 PUT(DOT1,LINE1),Z%,AND
1790 GOSUB 1880
1800 GOSUB 1970
1810 GOSUB 2040
1820 LOCATE 25,1:SCREEN 0,1:PRINT "Use arrow keys for moveme
nt ! F4 = pixel on  ! F5 = pixel off ! F6= Done        "
1830 GOSUB 2150
1840 GOTO 2580
1850 IF ARRAY%(FONDX)= 255 THEN RETURN
1860 LOCATE LL,LL1*5:IF ARRAY%(FONDX)< 95 THEN PRINT CHR$(AR
RAY%(FONDX)+32) ELSE PRINT CHR$(ARRAY%(FONDX)-2)
1870 LINE1 =(LL-1)*9 :DOT1 =(((LL1*8)+1)*5)-1
1880 '          this routine copies the character bytes from
the font table
1890 '          and pokes them into the specified screen memo
ry location
1900 IF ARRAY%(FONDX) = 255 THEN RETURN
1910 DEF SEG=(&HE000)
1920 GOSUB 2520
1930 FOR XXX=1 TO  9
1940 A=ARRAY%(FONDX+XXX)
1950 POKE (LIN+BYT+((XXX-1)*128)),A
1960 NEXT XXX:SCREEN 0,0:RETURN
1970 '          this routine puts the font matrix on the scre
en
1980 FOR COL = 0 TO 7:PUT (X+(COL*20),Y),Z%,OR:NEXT COL
1990 GET(400,35)-(560,51),Z3!
2000 FOR ROW = 1 TO 8:PUT(X,Y+(ROW*16)),Z3!:NEXT ROW
2010 RETURN
2020 '
2030 PRINT "FATAL LOAD ERROR -- ";ERROR1$;" FORMAT -- END OF
 "ERROR1$;" NOT FOUND":STOP
2040 '          this routine copies the character being exami
ned into the font
2050 '          matrix
2060 FOR ROW = 0 TO 8
2070 FOR COL = 0 TO 7
2080 IF (POINT(DOT1+COL,LINE1+ROW))>1 THEN PUT(X+(COL*20),Y+
(ROW*16)),Z1%,OR ELSE PUT(X+(COL*20),Y+(ROW*16)),Z%,AND
2090 NEXT COL:NEXT ROW
2100 PUT(CURSORX,CURSORY),Z2%:RETURN
2110 '          this routine drives the font matrix cursor
2120 '          cursorx & cursory point to cursor
2130 '          x offset would be +/- 20
2140 '          y offset would be +/- 16
2150 T$=INKEY$:IF LEN(T$) = 0 THEN GOTO 2150
2160 IF ASC(T$)> 27 AND ASC(T$) < 32 THEN GOSUB 2210
2170 IF T$="N" THEN GOSUB 2390
2180 IF T$="F" THEN GOSUB 2410
2190 IF T$="D" THEN RETURN
2200 GOTO 2150
2210 ARROW = (ASC(T$)-27)
2220 ON ARROW GOSUB 2240,2280,2310,2340
2230 RETURN
2240 IF CURSORX => 540 THEN RETURN
2250 GOSUB 2380
2260 CURSORX=CURSORX+20
2270 GOTO 2370
2280 IF CURSORX <= 400 THEN RETURN
2290 GOSUB 2380
2300 CURSORX=CURSORX-20:GOTO 2370
2310 IF CURSORY <= 49 THEN RETURN
2320 GOSUB 2380
2330 CURSORY=CURSORY-16:GOTO 2370
2340 IF CURSORY => 177 THEN RETURN
2350 GOSUB 2380
2360 CURSORY=CURSORY+16:GOTO 2370
2370 PUT (CURSORX,CURSORY),Z2%,PSET:RETURN
2380 PUT (CURSORX,CURSORY),Z2%:RETURN
2390 PUT (CURSORX,CURSORY-14),Z1%,OR
2400 PSET(DOT1+(CURSORX-400)/20,LINE1+(CURSORY-35)/16-1),7:R
ETURN
2410 PUT (CURSORX,CURSORY-14),Z%,AND
2420 PSET(DOT1+(CURSORX-400)/20,LINE1+(CURSORY-35)/16-1),0:R
ETURN
2430 GOSUB 1970 '       clear font
2440 FOR COUNT = 0 TO 8
2450 GOSUB 2490
2460 A=PEEK(LIN+BYT)
2470 LINE1=LINE1+1:NEXT COUNT
2480 GOTO 2570       ' clear 25th line
2490 '          screen position calculator
2500 '
2510 '          returns byt and lin as variables
2520 LIN1=(LINE1)\9:LIN2=(LINE1) MOD 9
2530 DT1=DOT1\8:DT2=DOT1 MOD 8
2540 LIN=LIN1*(&H800)+LIN2*(&H80)
2550 BYT=DT1:IF DT2 = 0 THEN ELSE BYT = BYT+1
2560 RETURN
2570 LOCATE 25,1:PRINT " "CHR$(27)+"l";:RETURN  'clear 25th
line !!
2580 '          this routine will save the newly created or m
odified
2590 '          font character into the font array and displa
y it on the screen
2600 '          in the display font table
2610 GOSUB 2570       '          clear 25th line
2620 LOCATE 25,1:PRINT "Save character in Font Array  Y/N --
>";:GOSUB 2770
2630 IF ANS=1 THEN 2640 ELSE 2740
2640 '          copy new character into font array
2650 LINE1= 99: DOT1 = 600
2660 GOSUB 2490
2680 FOR INDEX0 = 1 TO 9
2690 ARRAY%(FONDX+INDEX0) = PEEK((LIN+BYT)+((INDEX0-1)*128))
2700 NEXT INDEX0
2710 IF ARRAY%(FONDX) = 255 THEN ARRAY%(FONDX)=CAR ELSE 1520
2720 ARRAY%(FONDX+10)=255 '     put new end of font marker in
 array
2730 GOTO 1520
2740 GOSUB 2570       '          clear 25th line
2750 PRINT "EXIT BACK TO MAIN MENU Y/N -->";:GOSUB 2770
2760 IF ANS=1 THEN RETURN  ELSE GOTO 1520
2770 SS$=INPUT$(1):IF SS$="Y" OR SS$="y" THEN ANS = 1 ELSE I
F SS$="N" OR SS$="n" THEN ANS = 2 ELSE BEEP:GOTO 2770
2780 PRINT SS$;:RETURN
2790 IF ERR=53 THEN PRINT "NEW FILE Y/N ->";:GOSUB 2770
2800 IF ANS =2 THEN RESUME 200
2810 ARRAY%(0)=255     ' PUT END OF ARRAY CHARACTER IN ARRAY
2820 RESUME 500
```



Ashton-Tate

d BASE II*	$450
FPL	$450
Bottom Line Strategist*	$295
Friday	$225
Everyman's DB Primer	$14

*Now available under ZDOS.

Fox & Geller

QuickCode	$239
dUTIL	$69
dGRAPH (Z80)	$239

Humansoft

dBPLUS	$110

Terms: Some disk formats not available from original producer. In these cases we can copy to a compatible format for you for a $2/disk media charge. Add 2% shipping and handling plus 5 ⅛% sales tax if shipped to a Missouri address. Alaska, Hawaii, FPO's, APO's, and Canadian orders add 5% shipping. Prices above contain a 2% prepayment discount, add this 2% for approved purchase orders. Software Wizardry, Inc. also handles many other compatible products (both hardware and software). *Call or write for our FREE price list on these items.*

VISA

SOFTWARE WIZARDRY, INC.
122 YANKEE DRIVE ST. CHARLES, MO. 63301
(314) 946-1968

MasterCard

Eliminate the Limitations!

Recapture the Excitement of High Technology in your H/Z89-90

DG SUPER 89 ..... $829.00

The Super 89 **(Now including 128K Memory and the Super 89 Electronic Disk)** replaces the central processor board in the Heath/Zenith 89-90 series computers to bring your 89-90 to current state-of-art technology. The Super 89 transforms the Heath/Zenith 89-90 into a powerful, professional quality system meeting the needs of today and tomorrow. 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:

- Shipped with 128K memory
- Includes Super 89 Electronic Disk
- 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 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

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 spped; allows the use of "scratch pad" memory to increase efficiency; and provide for multi user capabilities.

Super 89 Electronic Disk

This software package for the Heath/Zenith CP/M 2.2.03 allows the Super 89 user to access auxiliary RAM as a very fast mass storage device. Provides up to 180K bytes (fully populated) of storage area that is accessible 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.

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 feature 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 supports full compatibility with the HDOS or CP/M disk operating systems. This feature gives you the best of both worlds in the amount of existing software you may use.

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 eliminate the limitations.

SUPPORT PRODUCTS for the SUPER 89

Arithmetic Processor $200.00

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.

Enhanced Super 89 Monitor $49.00

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.

MP/M II™ $550.00

MP/M II is a multi-user, multi-tasking operating system for use with the Super 89. Supports up to four users and the Z37, Z47 and Z67 disk systems. Compatible with CP/M software.

DG Super 89/128 K	829.00
DG Super 89/192 K	909.00
DG Super 89/256 K	989.00
Documentation Only	25.00

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.

Using the H/Z-29 Terminal

Pat Swayne
Software Engineer

Heath has come up with another "goodie" in the new H/Z-29 video display terminal. I have one connected to the H8 in my HUG office now, and I like it so much that I have just about retired my poor old H-19. However, it took a bit of effort to interface it to my system and get it working with all of my software. In this article, I will present my experiences with the H/Z-29 so that others thinking of getting one can easily get it "up and running". But first, I would like to explain some of the features of this terminal that make it so likeable.

H/Z-29 Features

Probably the first thing you notice about the H/Z-29 is that it has a detachable keyboard. It is the first product from Heath/Zenith to have one (unless you count the ZT-1). That means that not only can you type with the keyboard in your lap, but if you prefer to keep the keyboard on the desk, you can put the screen where you want it. I know that some people prefer not to have it so close to their face as it is with fixed keyboard terminals.

The keys themselves are made in the so-called "sculptured" design. My first experience with that kind of keyboard was on an Intel 200-series Microcomputer Development System, about three years ago, and I hated it. I guess they have learned something about keyboards since then, because this keyboard is a joy to use. The keys are easier to press than H/Z-19 or H/Z-89 keys, and a fast typist can really fly on the H/Z-29. Pressing several keys at once does not cause erroneous responses as on the H/Z-19.

The numerical keypad has double size zero and ENTER keys, and a comma key in addition to the usual H/Z-19 keys. There are arrow and HOME keys on the main keyboard, but the 2, 4, 5, 6, and 8 keys can also serve as arrow and HOME keys when the keypad is shifted. There are two new function keys not found on the H/Z-19 or the H/Z-89. The F9 key, which produces ESC 0 I, and the HELP key, which produces ESC ^ (tilde). The blue, red, and white keys are replaced with F6, F7, and F8, and the ERASE key is at the right end of the top row, instead of in the middle of the row as on the H/Z-19.

All of the things that you would normally have to be set up using switches inside the case, or escape codes typed in while OFF LINE, can be set on the H/Z-29 keyboard via a special Set Up mode. A SET UP switch located at the left end of the top row, toggles between the set up and operating modes. Set up menus, which appear on the 25th screen line, are accessible by typing the letters A through G and T (for the Tab menu) while you are in the set up mode. The individual items in each menu are altered by typing a number listed beside the item. For example, to change the baud rate, you press B to select the appropriate menu, then press the number 1 until the rate desired appears on the 25th line. The rate displayed increments once each time you press the 1 key until the highest rate (19200) is reached, and it starts over with the lowest (75). Tab stops can be set anywhere you want them when you select the Tab menu, by moving a cursor on the 25th line (filled with numbers when you press T) with the arrow keys and pressing the up arrow to set a tab, or the down arrow to clear one. Of course, you should bear in mind that these tab stop settings affect only what appears on the screen. You would have to reset the tab stops on your printer (if they can be set) to make a printout of your text look like it did on the screen, unless you use standard settings. And remember that in CP/M, tabs are expanded to spaces when a file is TYPEd to the screen.

Once you have set things the way you want them, you can press the SET UP key again to temporarily record the settings, or SHIFT-SET UP to record them permanently. On this last point, there was a slight boo-boo in the manual in that the only place where it said that SHIFT-SET UP permanently recorded set up changes was in the discussion of the tab settings. I scratched my head a little until I found that. However, that should be corrected soon, if it has not been done already.

The H/Z-29 has several new video features that you will probably have fun playing with. There are several video attributes that you can set, including half intensity, underline, and blinking. There is also an alternate character set that inlcudes sub- and super-script characters and new graphic characters. All of the attributes and special characters can be selected on a character by character basis. For those of you who already have or will soon get an H/Z-29, here is a BASIC program that illustrates the attributes and alternate characters.

```
10 DIM B$(15)
20 E$=CHR$(27):S$=E$+"s":N$=S$+"0":A$=S$+CHR$(64)
30 FOR I=0 TO 15:READ B$(I):NEXT I
40 PRINT E$;"E":PRINT
50 PRINT "HERE ARE THE H29 VIDEO ATTRIBUTES:"
                                        :PRINT:PRINT
60 FOR I=0 TO 15
70 PRINT "THIS IS ATTRIBUTE NO. ";
                                  S$;CHR$(I+48);B$(I);N$;
80 IF I/2<>INT(I/2) THEN PRINT :GOTO 100
90 PRINT TAB(40);
100 NEXT I
110 PRINT :PRINT A$;"THESE";N$;" ";A$;"ARE";N$;" ";
120 PRINT A$;"alternate";N$;" ";A$;
                          "CHARACTERS";N$;".":PRINT
130 DATA ZERO,ONE,TWO,THREE,FOUR,FIVE,
                                SIX,SEVEN,EIGHT,NINE
140 DATA TEN,ELEVEN,TWELVE,THIRTEEN,FOURTEEN,FIFTEEN
```

☞

This program was written to run under just about any version of BASIC, including MBASIC, Benton Harbor BASIC, and BASIC-E. Notice in line 110 that we must switch to normal characters (with N$) to print spaces between words because the space produces a graphic character while you are in the alternate set.

Some of the new features on the H/Z-29 are available only through ANSI escape codes. Among these are the blinking rate (when the blinking attribute is set) and setting the screen clock. The screen clock cannot be read from an escape code, which is a bit of a disappointment. An engineer who was involved in the design of the H/Z-29 told me that they ran out of ROM space to add everything they wanted. The code is in (2) 4K ROMs. Bigger ROMs could be used, so perhaps there will be a HUG ROM for the H/Z-29 in the future.

The processor in the H/Z-29 is an 8051 running at about 7 MHz. The 8051 is an improved version of Intel's 8048 controller processor family. There are not one but two of Intel's new video controller chips. One puts the characters on the screen and the other controls the attributes.

One of the nicest things about the H/Z-29 is the complete absence of "video tearing" on the screen during rapid screen writing. If you do not know what video tearing is, run this program on your H/Z-19 or H/Z-89 (in line 30, there are 40 spaces between the quotes).

```
10 E$=CHR$(27):PRINT E$;"E"
20 FOR I=1 TO 10:PRINT E$;"p";
30 PRINT "                                        ";
40 PRINT E$;"q":NEXT I:PRINT
50 FOR I=1 TO 100
60 PRINT "SCREEN TEARING DEMONSTRATION";E$;"A"
70 NEXT I:PRINT
```

The program will print a white (reverse video) rectangle on your screen and then print the message in line 60 100 times on the same line. During that time, tearing will be visible in the white area. On the H/Z-29, you can hardly tell that this program is running, and with video games, the creatures move cleanly with no tearing at all.

There has been some apprehension expressed that the H/Z-29 is slower than the H/Z-19, so you have to run it at a slower baud rate. In a way, that is true. The H/Z-29 takes longer to process escape sequences and certain other characters, and at high baud rates it can fall behind the incoming data with the result that "garbage" appears on the screen. However, provision is made for either hardware (CTS) or software (XON-XOFF) handshaking so that the H/Z-29 can signal the computer to wait until it catches up. With proper handshaking, you can run the H/Z-29 at full speed (19,200 baud) and there is no visible slowing of the display (compared to the H/Z-19) when you play video games, etc.

Connecting the H/Z-29 to Your Computer

If you use only HDOS (not CP/M), you will be happy to know that HDOS supports XON-XOFF handshaking, so all you have to do is set up the H/Z-29 for software handshaking, connect it to your computer, and go. At least, I thought that was all you had to do, but it didn't work when I first tried it. It seems that HDOS sets up pin 5 (RTS) of the RS-232 lines to the wrong polarity. So I opened up the shell on one end of my RS-232 cable, clipped the wire going to pin 5, and presto! It worked. If you have one of the new molded RS-232 cables, break off pin 5 at the male end of the cable, or clip the wire going to pin 5 inside the H/Z-29 cabinet. Pin 5 does not seem to be required for normal operation with either hardware or software handshaking.

If you use CP/M, you must patch your BIOS to use hardware handshaking since CP/M does not support XON-XOFF and since patching in hardware handshaking is easier than patching in XON- XOFF. If you use both HDOS and CP/M, you should patch both to use hardware handshaking to avoid having to switch handshaking modes when you switch operating systems. I have worked out patches for both operating systems to support hardware handshaking.

Patching HDOS

The following patch will convert HDOS 2.0 to hardware handshaking with the terminal. The patch should be made using the program PATCH.ABS that is supplied with HDOS.

```
>PATCH

PATCH Issue #50.06.00

File Name? HDOS.SYS
Patch ID? IFOJIC
Prerequisite Code? IFBEIADPGEFFCF

Address? 20127
020127 = 146/131
020130 = 071/            (just hit RETURN
020131 = 333/                  where no entry is shown)
020132 = 373/356
020133 = 346/
020134 = 001/020
020135 = 312/
020136 = 131/
020137 = 071/
020140 = 361/000
020141 = 323/000
020142 = 372/000
020143 = 303/
020144 = 160/146
020145 = 071/^D          (Control-D typed)
Address? ^D
Patch Check Code? GPEGPNNB

PATCH Issue #50.06.00

File Name? ^D
```

After this patch is made, HDOS will handshake properly with the H/Z-29. This patch overwrites the part that allows HDOS to work with the old H8-5 interface card. It does not disable software handshaking, so that will still work.

Here is the code that is patched (part of the routine SCOUT in HDOS.SYS).

```
OLD

SCOUT9  LDA     S.CDB
        CPI     1       ;H8-4?
        JZ      SCOUT92
SCOUT91 IN      373Q    ;H8-5 PORT
        ANI     1
        JZ      SCOUT91
        POP     PSW
        OUT     372Q

        JMP     SCOUT95
SCOUT92 IN      355Q

NEW

SCOUT9  LDA     S.CDB
        CPI     1
        JZ      SCOUT91
SCOUT91 IN      356Q    ;H8 PORT
        ANI     20Q
        JZ      SCOUT91
        NOP
        NOP
```

```
        NOP
        JMP     SCOUT92
SCOUT92 IN      355Q
```

Patching CP/M

The following patch will convert CP/M 2.2.03 to hardware handshaking. This patch is a bit more difficult to implement than the HDOS patch due to the fact that the location of the patch area may not be in the same place in everyone's BIOS, and so must be calculated. When you see "xxxx", it refers to a 4 digit hex number whose value is unimportant to the patch. Any other combination of lower case letters, such as "bbbb", refers to a 4 digit hex number that will be used later, and so must be remembered. This patch is only valid if the CRT routines in the BIOS have not been modified.

After the patch is made, reboot the system to put the new BIOS into memory. CP/M will now handshake properly with an H/Z-29 with hardware handshaking set. As with the HDOS patch, this one overwrites code used to communicate with the H8-5 card. You must use an H8-4 card or equivalent in an H8 to use this patch.

What About the H/Z-89?

If you would like to use an H/Z-29 with an H/Z-89 computer, there are two approaches you can take. One is to remove the cable that goes to the Terminal Logic Board from P513 on your CPU board, connect a short RS-232 cable (Heath P/N 134-1073) to P513, and connect the H/Z-29 to the other end. Another approach might be to replace the Terminal Logic Board and keyboard in an H/Z-89 with the ones from an H/Z-29. To my knowledge, that has not been tried, and would probably be quite a project. If I ever try it, I will write about it in REMark.


```
A>DDT
DDT VERS 2.2
-L0,2
  0000  JMP  aaa3
  0003
```
Locate the address of the BIOS warm boot entry point in your system. It will always end with the number 3, and so we have shown it here as "aaa3".

```
-L18,1A
  0018  JMP  bbbb
  001B
```
Locate the address of the CRT interrupt service routine in your system, indicated here as "bbbb".

```
-Hbbbb,aaa0
xxxx cccc
```
Find the offset from the start of the BIOS to the CRT interrupt service routine by subtracting the above two addresses. Replace the 3 in the warm boot address with a zero before subtracting.

```
-Hcccc,10A
dddd xxxx
```
Find the offset from the start of the BIOS to the CRT output status routine, which is 10A (hex) bytes above the CRT interrupt service routine.

```
-Hdddd,200
eeee xxxx
```
Find the actual address of where the CRT status routine will be when you load the BIOS into memory. Jot down the result (eeee) for later use.

```
-IBIOS.SYS
-R
NEXT  PC
yyzz 0100
```
Load your BIOS into memory. Jot down the number "yyzz" for later use.

```
-Leeee
  eeee  LXI  H,0037
  xxxx  LXI  D,xxxx
  ffff  LDA  0036
  xxxx  RAR
  xxxx  JC   gggg
  xxxx  CALL xxxx
  xxxx  JZ   xxxx
  xxxx  LDAX D
  xxxx  ORA  A
  xxxx  JNZ  xxxx
  xxxx  DCR  A
```
Disassemble the BIOS at the calculated CRT output status routine address (eeee) to make sure we are at the right point. Note the addresses marked with characters other than "xxxx". Do not make the patch if this disassembly does not show what we have shown here.

```
-Hffff,200
xxxx hhhh
-Hgggg,200
iiii xxxx
```
Subtract 200 (hex) from address ffff, and add 200 to address gggg. Save the results for later.

```
-Affff
ffff  CALL gggg
xxxx  NOP
xxxx  JZ hhhh
xxxx  .
```
Insert the first patch at address ffff.

(Type a period.)

```
-Aiiii
iiii  IN EE
xxxx  NOP
xxxx  LXI H,37
xxxx  ANI 10
xxxx  RET
xxxx  .
-^C
```
Insert the second patch at iiii.

(Type a period.)

(Type Control-C.)

```
A>SAVE nn BIOS.NEW
```
Save the patched BIOS on your disk. Use the number "yyzz" you jotted down earlier to calculate "nn" as follows: If "zz" is 00, convert "yy" to decimal and subtract one to get "nn". If "zz" is not zero, convert "yy" to decimal but do not subtract one.

```
A>STAT BIOS.SYS $R/W
```
Remove the R/O attribute from the old BIOS.

```
A>REN BIOS.OLD=BIOS.SYS
```
Rename the old BIOS.

```
A>REN BIOS.SYS=BIOS.NEW
```
Replace it with the new one.

```
A>STAT BIOS.SYS $R/O
```
Set the R/O attribute.

```
A>STAT BIOS.SYS $SYS
```
Set the SYS attribute.

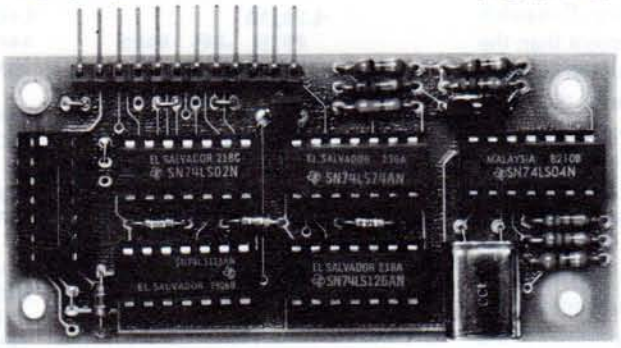
MORE SOLUTIONS

from KRES

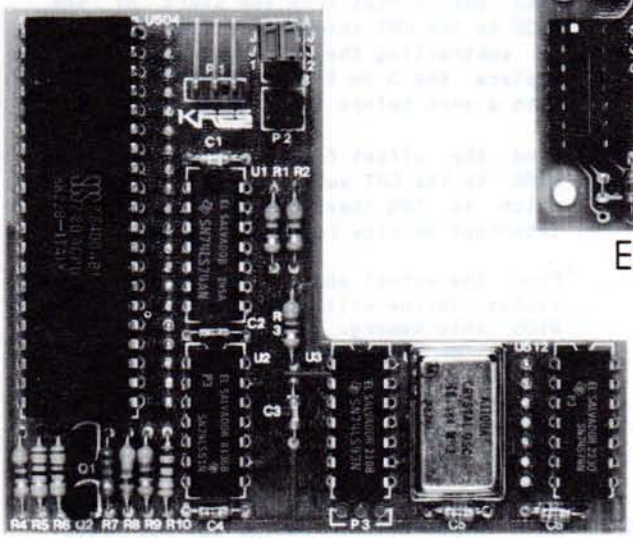
the DUAL SPEED MODULE

a faster solution

Now Available in Two Versions



ESM-240 for the Expansion



DSM-240 for the H/Z 89-90

Why run half speed when you can run full speed? Install one of these KRES Modules in your H/Z 89-90 and cut computation time in half. The supplied HDOS or CP/M software will allow you to operate at regular or double speed (2 or 4 MHZ).

Either version with software $79.95

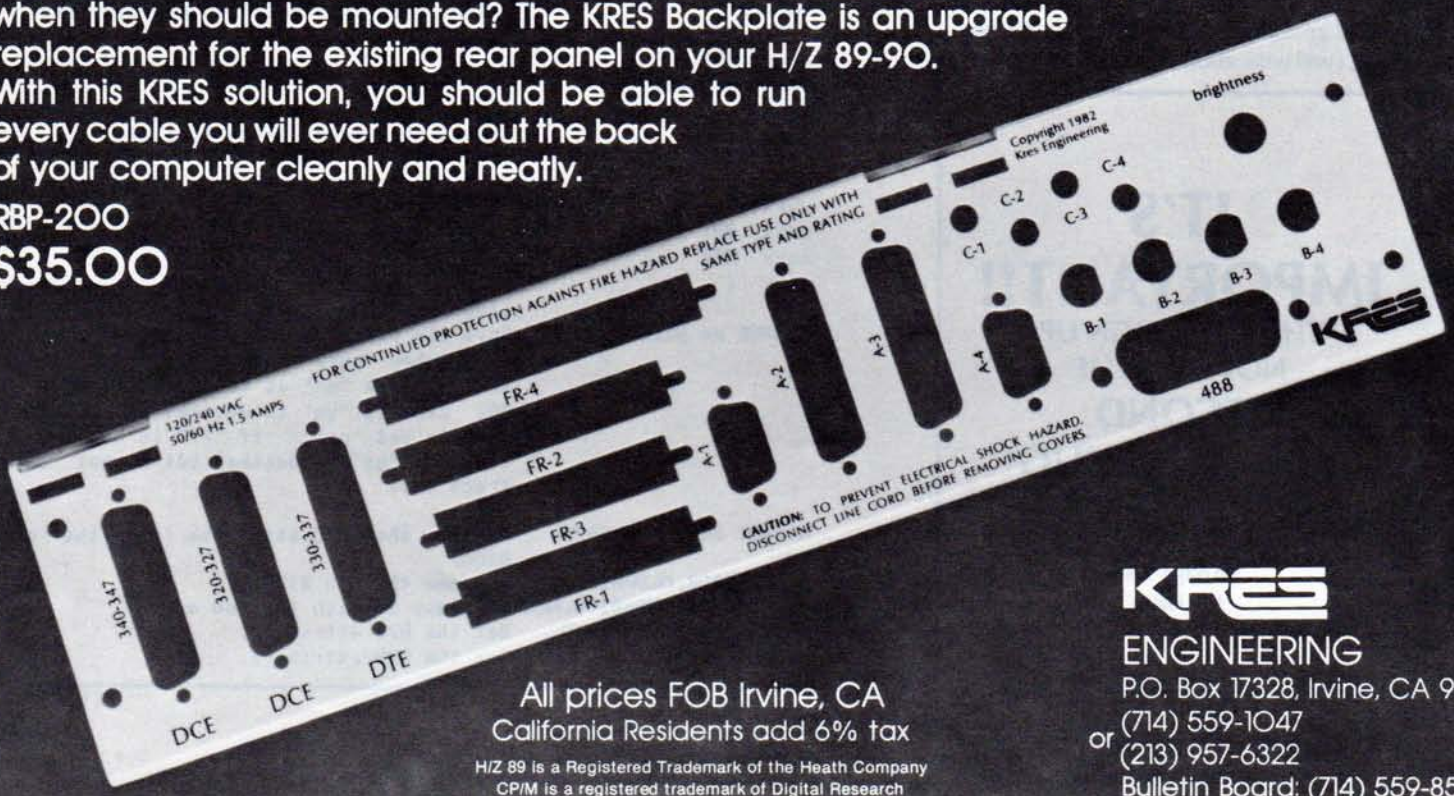
the BACKPLATE

a neater solution

Are you tired of having wires and cables stuffed out the holes of your computer when they should be mounted? The KRES Backplate is an upgrade replacement for the existing rear panel on your H/Z 89-90. With this KRES solution, you should be able to run every cable you will ever need out the back of your computer cleanly and neatly.

RBP-200

$35.00



All prices FOB Irvine, CA

California Residents add 6% tax

H/Z 89 is a Registered Trademark of the Heath Company

CP/M is a registered trademark of Digital Research

KRES ENGINEERING

P.O. Box 17328, Irvine, CA 92713

or (714) 559-1047

(213) 957-6322

Bulletin Board: (714) 559-8579

The Wand vs. The Star

A Review of Two Top-of-the-Line Word Processing Packages

John Immerwahr
Philosophy Department
Villanova University
Villanova, Pa 19085

For those of us who are neither number crunchers nor computer game freaks, word processing is one of the applications that can make a computer really pay for itself. For anyone who does a significant amount of writing, the most obvious benefit is that word processing makes it possible to eliminate much of the expense and delay of having a professional typist retype text into clean copy. Changes and corrections are so easy to make that even sloppy typists can easily produce beautiful final products. An even more important advantage is that many writers also find that they compose more quickly and more fluently with word processing. Many professional writers use it as an aid to the composition process itself, rather than merely as a way of eliminating secretarial expenses.

There are a wide variety of word processing programs available for Heath/Zenith computers, including inexpensive but serviceable programs such as Software Toolworks' combination of PIE for editing and TEXT for printing (total cost of about $70). But serious writers will also want to look at the two top-of-the-line word processing packages offered, Micropro's WordStar ($395) and Peachtree's Magic Wand ($295). These programs (which require the CP/M operating system) are both outstanding pieces of software. Magic Wand has a capability for doing form letters with individualized addresses. WordStar can do this even better, but only if you buy the optional Mailmerge program ($135).

One thing should be clear from the start. WordStar is a much more powerful program than Magic Wand. There is nothing that Magic Wand can do that can't be done in one way or another with WordStar, and WordStar has many capabilities that Magic Wand cannot duplicate. But as in all things, sheer power should not be your only criterion for selecting an expensive piece of software. For many users, the less ambitious Magic Wand may be equally good or even superior. In the final analysis, your choice will depend on your needs, and on your pocketbook. Here is a point-by-point comparison:

1. Documentation. Let's start by looking at the manuals that come with these programs. Both come in handsome black and silver looseleaf binders, but the resemblance ends there. The Magic Wand manual is laid out in beautifully written exercises that introduce you to the program in a logical step-by-step manner comprehensible even to someone who knows nothing about computers or word processing. Each step is illustrated with examples, and sample lessons are included on the disks. In addition to the lessons, there is also a more formal reference manual where experienced users can check on specific points.

The WordStar manual, on the other hand, is an example of the kind of writing that has given computers a bad name among the uninitiated. Basically it is a reference document, useful for an experienced programmer but virtually worthless to the novice. Fortunately, WordStar is such a popular program that independently published manuals are also available. My favorite is INTRODUCTION TO WORDSTAR by Arthur Naiman (Sybex, 1982— about $12). This is a nice breezy presentation that walks you through WordStar and compensates for the fact that no suitable manual comes with the package.

2. Compatibility with Heath/Zenith 89. One of the nice things about the H/Z-89 is its compliment of extra keys. In addition to the usual keys, it has a number keypad and nine special function keys. These keys make the H/Z-89 an excellent choice for word processing. The most important feature in word processing is speed and ease of use. Writing is very time consuming, and anything that can speed up the process is a real plus. Having special keys with unique functions can dramatically increase efficiency.

Magic Wand is beautifully married to the H/Z-89 keyboard. Casual inspection of the keyboard indicates that the number keys on the keypad already have additional markings (arrows on some and letters on others). Magic Wand assigns functions to these keys that correspond to the existing markings; the arrow keys move the cursor in the directions indicated, and the IC key, for example, activates the Insert Character mode. Magic Wand also assigns functions to all of the special function keys. By happy coincidence, all but two of the Magic Wand commands are accommodated by one of the existing keys.

Astoundingly, WordStar makes no use of these keys whatsoever. Instead the commands must be executed by pushing two keys at once, the control key and some other key. Instead of using the cursor arrow keys on the keypad, for example, the writer must push the control key and the S key to move the cursor to the left, or push the control key and the D key to move the cursor right.

As with the lack of a good manual, this problem can be corrected. An enterprising independent dealer named Patrick McNally (PO Box 578, Haleiwa, Hawaii, 96712) sells a neat little package called 89/Star. This program automatically changes WordStar so that it

takes advantage of all of the numberpad keys and the special function keys. McNally's program is very easy to install and costs about $20. In addition to assigning WordStar commands to all of the special function keys, 89/Star even writes labels for the keys on the bottom line of the screen.

Magic Wand is also much more modest in its requirements for hardware. The program is designed to make life easy for someone who only has one disk drive, and 48K is plenty of memory. Because of its complexity, WordStar requires a great deal more disk space and more RAM. You could write short documents with only one disk drive, but it would be extremely inconvenient to do so, and 64k is strongly recommended.

3. Ease of Learning. When it comes to ease of learning, the advantages are all on the side of Magic Wand. The explanation for this is simple, since Magic Wand can do less, there is less you have to learn. Given its tremendous power, WordStar necessarily has many more commands and options. WordStar tries to compensate for this with an impressive system of menus. The instructions for what to do actually appear at the top of the screen. If you want to move some text around, the menu tells you to press control and K. When you do that a new menu appears that gives the commands for many different options. Theoretically you don't have to remember anything in WordStar, because the system is always producing a new menu for you that tells you what your choices are. But in the final analysis, complicated is complicated, and you can only be really efficient when you don't have to read about a command each time you use it. Most people will become proficient much sooner with the Wand.

4. Editing. By now you've probably gotten the idea that Magic Wand is a lot easier to get started with. But in the long run, this is less important than it might seem. Neither system is really that difficult to learn, and in a few days or weeks you will be really proficient with either one. So don't be put off by the difficulties in getting started with WordStar - think about the long run too.

Once you have mastered the system, WordStar's power really begins to pay off. For example, one of the most important things in word processing is moving the cursor around. Nothing slows you down more than having to bang around trying to get where you want to be. Suppose, for example, that you want to move the cursor to make a change at the right hand end of a line of text. Magic Wand lets you move the cursor to the right one space at a time. If you get bored, you can hold the repeat key down, but you are still only going one space at a time. The Star lets you move one space at a time, one word at a time, and there is even a command to move immediately to the right hand end of the line.

The other commands are much the same. Like all word processing programs, Magic Wand lets you search automatically for a string (a word or phrase) and, if you wish, change it to something else. This is a real advantage in making corrections. Rather than hunting around for the place where you want to change a misspelling, for example, search/replace will find the place for you and make the change all at once. WordStar does this too, of course, but it gives you many more options as to how to do it. You can search forward or backward in the document (only forward in the Wand), you can have the changes made automatically or the program will ask you about each one (only automatic in the Wand) and you can tell WordStar not to distinguish between upper and lower case. Suppose, in other words, that you want to change every occurence of the word "college" to "university." Magic Wand will only look for "college" but will ignore "College". The Star can be instructed to look for both. Generally wherever Magic Wand has one way to do something, WordStar will have several. Once you master these commands and options, they give you real power and speed.

5. Printing. Here again, the advantages are with WordStar. The big problem with Magic Wand is that you never really know what your document is going to look like on the page until you actually print it out. Since printing can be time consuming, this is a real disadvantage. For example, you don't want a chart broken between two pages, or you don't want a section title to be the last line on a page. Magic Wand gives you commands to control for some of these things, but ultimately there is a lot of guesswork involved. There is a way to preview the text on the screen, but even when you do this you only get an approximation of what you will actually see when your document is printed out.

The real strength of WordStar is that there are no surprises. You see exactly what you are going to get on the printed page. If your paper is double-spaced, you will see it double-spaced on the screen. If your document is right-justified, you will see it justified on the screen. You always know exactly where the page breaks will fall, so there is never a danger of breaking a chart in two pieces, or leaving a straggling line at the top or bottom of the page. This is a tremendous advantage whenever you write a document where the format is important. For anyone who is producing final copy that has to look just right, this alone would dictate the choice of WordStar over Magic Wand.

For doing "quick and dirty drafts", however, Magic Wand has some advantages too. First it prints out more quickly, and it also has a helpful quick-print feature that lets you knock out a simply formatted hard copy without even saving your text to disk. Magic Wand also allows you to examine and pull in pieces of other texts that you have already written (WordStar only allows you to pull in the whole document, rather than a section of it).

6. Compatibility with the Epson MX-80 printer. The Epson MX-80 is one of the more popular dot matrix printers. One of its strengths is that it has a variety of print formats, plus a set of graphics characters. It is possible to use these features with Magic Wand, but you have to read the manuals rather carefully to find out exactly how to do it. The secret is the OUT command, issued either in the text or at print time, followed by certain printer codes, will turn on or off the different print modes. The codes are all given in the Epson Manual as statements in BASIC, but it is easy to translate them into commands Magic Wand understands. LPRINT CHR$(27) CHR$(69) is the BASIC command to turn on intense type. OUT 27,69 in Magic Wand accomplishes the same thing. You can even use the OUT command to put graphic characters in your text.

The distributed version of WordStar does allow you to double print characters, but it does not seem to be possible to produce any of the other Epson print modes. Here again Patrick McNally comes to the rescue with Ep/Star. This is another program to modify WordStar so that it can use the special Epson print modes. Turning these modes on and off is cumbersome in Magic Wand but you don't have to buy a special program to do it. Once you do have the modification, it is much easier to use these special features from WordStar (but if you bought both programs, the manual, and MailMerge, WordStar would cost you $287 more than Magic Wand).

7. Compatiblity with other computers. The complexities of WordStar create problems if you want to interface with other word processing systems. I frequently produce texts which I send over the telephone to a dedicated wordprocessor in another city. As

it turns out, I can only send Magic Wand files. The reasons for this are a bit technical, and I won't explain them in detail. Some word processors use the eighth bit as a way of distinguishing between hard returns and soft returns, and hard spaces and soft spaces. Some modem transfer programs don't even transmit the eighth bit, and few other word processing programs will recognize these characters. Magic Wand, on the other hand, produces a pure ASCII file that can be easily understood by almost any other program (including WordStar).

Overall Assessment

So which one is right for you? As usual, the answer will depend on your needs. If you are doing longer documents or drafts, where formatting is not that crucial, and if you want a cheaper, faster, and easier system, Magic Wand is for you. I happen to own both. I started out with Magic Wand but then got WordStar to be compatible with co-authors who were using it. I prefer to use Magic Wand when I am doing a draft version of an article that has few charts or tables. But when I need to do something like a resume, or a proposal that must look perfect, I immediately fire up WordStar.

About the Author:

John Immerwahr is a Professor of Philosophy at Villanova University, and a Senior Project Consultant for the Public Agenda Foundation. He uses his computer primarily for word processing and for writing educational software for his pre-school children.

Personal Checks and Balancing

Don Thomas
3797 Southfield Dr.
St. Joseph, MI 49085

The programs CHECK.BAS and BALANCE.BAS are all written in a user friendly, self-prompting style. The check book program for example prints out as a standard checkbook register does. The programs were originally written for my wife as I was learning BASIC programming. All of my family check records are now kept on the computer. Note that in all of my programs I use a default no-return unless otherwise prompted.

CHECK.BAS is more easily managed using monthly data files so the first prompt asks the month's entries to be used, such as JAN. The program will add the .DAT extension. Using 100K drives, I used SY1: for my data files, however, you are prompted for the drive you wish to use for data storage. The program will ask if you want to initialize a data file. Be careful when answering 'Y' to this as it is used when starting new monthly data files on the disk. Old files may be renamed from HDOS such as JAN1983.DAT for permanent storage. The program asks if you wish to add entries to the checkbook. You may either add now or after you see the register printed out. An editor, of sorts, is present to remove the last line entered in case of a mistake. You may now make more entries or get a hard copy printout of what you have. This printout is in standard checkbook form.

BALANCE.BAS reconciles your checkbook register to the bank's monthly statement. You are first asked to enter how much the bank thinks you have in your account. Then you are asked to enter your outstanding deposits. Enter them one at a time until finished, then just hit a return to continue with the program. Entering outstanding checks is done in the same way as the deposits, but you also enter the check number. When you are done entering checks, the computer will show you your adjusted balance. If the balance doesn't agree, the computer will then help you find your mistake by showing you the difference between your figures and the bank's figures. It can also read back your outstanding checks and deposits.

CHECK.BAS

```
10 '          WRITTEN BY    Don Thomas
20 '                        3797 Southfield Dr.
30 '                        St. Joseph, Mi.
40 '                        (616) 429-6382
50 CLEAR 3000
60 ON ERROR GOTO 1230
70 DEFDBL B,C,D
80 DIM S$(100)
90 F$=
   "\     \ \      \   \                    \ ####.##   ####.##   ####.##"
100 PRINT CHR$(27)+"E"
110 C2$=C1$
130 LINE INPUT "Which month's entries do you wish to use? ";C1$
140 IF C1$="" THEN C1$=C2$
145 O=LEN(C1$)
150 PRINT :PRINT "Which drive has the data files? <1> 0 or 2?";:P$=INPUT$(1)
160 PRINT "  ";P$
170 IF P$="1" OR P$=CHR$(13) THEN DR$="SY1:"
180 IF P$="0" THEN DR$="SY0:"
190 IF P$="2" THEN DR$="SY2:"
195 'TBL SHOOT'PRINT C1$
200 C1$=DR$+C1$+".DAT"
205 'TBL SHOOT'PRINT C1$
210 M=LEN(C1$):IF M>16 THEN C1$=MID$(C1$,5,O)
                                        'SET C1$ ACCORDING TO FILE LENGTH
215 'TBL SHOOT'PRINT C1$
220 PRINT
230 PRINT
240 LINE INPUT "Did you mount the drive you wish to use? <Y> ";MD$
250 MD$=LEFT$(MD$,1):IF MD$="N" OR MD$="n" THEN GOSUB 1310
260 PRINT
270 LINE INPUT "Do you wish to initilize the data file? ";A1$
280 PRINT
```

```
290 A1$=LEFT$(A1$,1): IF A1$="Y" THEN 1170
300 LINE INPUT "Do you wish to add entries to the check book? ";A1$
310 A1$=LEFT$(A1$,1):IF A1$<>"Y" THEN 620
320 REM OPEN FILE TO READ IT
330 PRINT CHR$(27)+"["
340 OPEN "I",#1,C1$
350 FOR J=1 TO 100: INPUT #1,S$(J)
360 IF S$(J)<>"EOF" THEN NEXT J
370 CLOSE #1
380 PRINT CHR$(27)+"\"
390 J=J-1:M1$=RIGHT$(S$(J),10):M1=VAL(M1$):LET D=M1
400 J=J+1
410 B2$=" ":B3$=" "
420 B=0:C=0
430 LINE INPUT "Is this a deposit entry? ";A1$
440 A1$=LEFT$(A1$,1):IF A1$="Y" OR A1$="y" THEN 500
450 LINE INPUT "Check No. ";A$
460 LINE INPUT "Date check written ";B2$
470 LINE INPUT "Description of check ";B3$
480 INPUT "Amt. check written for ";B
490 GOTO 520
500 A$="****":INPUT "Amt. of Deposit ";C
510 LINE INPUT "Date of Deposit. ";B2$
520 D=D-B:D=D+C
530 ' WRITE OLD INFO AND NEW TO DISK
540 OPEN "O",#1,C1$
550 IF J<>1 THEN FOR I= 1 TO J-1
560 PRINT #1,S$(I)
570 NEXT I
580 PRINT #1,;USING F$;A$,B2$,B3$,B,C,D
590 PRINT #1,"EOF"
600 CLOSE #1
610 ' READ DISK AND DISPLAY ON TUBE
620 OPEN "I",#1,C1$
630 PRINT CHR$(27)+"E"
640 PRINT CHR$(27)+"["
650 FOR J=1 TO 100
660 INPUT #1,S$(J)
670 IF S$(J)="EOF" THEN 700
680 PRINT S$(J)
690 NEXT J
700 CLOSE #1
710 PRINT CHR$(27)+"\"
720 J=J-1:M1$=RIGHT$(S$(J),10):M1=VAL(M1$):D=M1
730 J=J+1: B2$=" ":B3$=" "
740 PRINT:PRINT
750 LINE INPUT "Do you wish to correct the last entry? ";A1$
760 A1$=LEFT$(A1$,1):IF A1$="Y" OR A1$="y" THEN 1030
770 PRINT:PRINT
780 LINE INPUT "Do you have anymore entries? ";A1$
790 A1$=LEFT$(A1$,1):IF A1$="Y" OR A1$="y" THEN 340
800 PRINT:PRINT"NOTE: If you require a printout, LP: must be loaded. "
810 LINE INPUT "Do you wish a printout? ";A1$
820 A1$=LEFT$(A1$,1):IF A1$="Y" OR A1$="y" THEN 900 ELSE GOTO 1210
830 REM DATA FILE INIT. SUBROUTINE
840 OPEN "O",#1,C1$
850 PRINT #1,;"NO.     DATE      DESCRIPTION            CHK.AMT.  DEP.AMT."
"   BALANCE"
860 PRINT #1,;" "
870 PRINT #1,"EOF"
880 CLOSE #1
890 GOTO 300
900 OPEN "I",#1,C1$
910 FOR J=1 TO 100:INPUT #1,S$(J):IF S$(J)="EOF" THEN 930
920 NEXT J
930 CLOSE #1
940 OPEN "O",#1,"LP:"
950 FOR J=1 TO 100
960 IF S$(J)="EOF" THEN 1000
970 PRINT #1,; S$(J)
980 NEXT J:CLOSE
990 PRINT
1000 CLOSE:PRINT:PRINT"Do you wish to return to the operating system? ";
1010 AA$=INPUT$(1):PRINT AA$:IF AA$="Y" OR AA$="y" THEN SYSTEM ELSE 10
1020 REM READ IN TO REMOVE LAST ENTRY
1030 OPEN "I",#1,C1$
1040 FOR J=1 TO 100
1050 INPUT #1,S$(J)
1060 IF S$(J)="EOF" THEN 1080
1070 NEXT J
1080 CLOSE #1
1090 REM GETS RID OF LAST ENTRY IF DESIRED
1100 J=J-1
1110 OPEN "O",#1,C1$
1120 IF J<>1 THEN FOR I=1 TO J-1
1130 PRINT #1,S$(I)
1140 NEXT I
1150 PRINT #1,"EOF"
1160 CLOSE #1:GOTO 610
1170 PRINT
1180 PRINT "CAUTION!!!!  ARE YOU SURE??? THIS REMOVES ALL DATA FROM SPECIFIED"
" FILE!! <N> ";
1190 'THE MIDDLE STATMENT ON THE NEXT LINE IS A LINE FEED CAUSE NO
      AUTO LINE FEED AFTER THE INPUT STATMENT. (CAUSES THE NEXT LINE
      TO BE PRINTED ON SAME LINE).(LOOK AT END OF 1010)
1200 A$=INPUT$(1):PRINT A$:PRINT CHR$(27);CHR$(10);
IF A$="Y" OR A$="y" THEN 840 ELSE 300
1210 PRINT :PRINT "Do you wish to return to the operating system?";
1220 X$=INPUT$(1):IF X$="Y" OR X$="y"THEN SYSTEM ELSE 100
1230 PRINT:IF ERR=53 THEN PRINT "File not located, do you wish to see the "
    "directory? ";:H$=INPUT$(1):IF H$="Y" OR H$="y" THEN 1330
1240 IF ERR=65 THEN PRINT"Hit return and type in something stupid!";
1250 IF ERR=57 THEN PRINT "Load lp: then rerun program ":SYSTEM
1260 IF ERR=57 OR ERR=65 OR ERR=53 THEN 1280
1270 ON ERROR GOTO 0
1280 IF H$=CHR$(13) OR H$="N" OR H$="n" THEN 1300
1290 C$=INPUT$(1)
1300 RESUME 10
1310 PRINT:PRINT:RESET DR$
1320 RETURN
1330 PRINT CHR$(13):PRINT:IF DR$="SY0:" THEN FILES
1340 IF DR$="SY1:" THEN FILES "SY1"
1350 IF DR$="SY2:" THEN FILES "SY2"
1360 PRINT:PRINT "Hit return to continue ";:GOTO 1250
```

```
NO.     DATE      DESCRIPTION            CHK.AMT.  DEP.AMT.   BALANCE

****    01-JAN                             0.00    532.00     532.00
201     02-JAN    RENT                   200.00      0.00     332.00
202     03-JAN    CAR PAYMENT             20.44      0.00     311.56
EOF
NO.     DATE      DESCRIPTION            CHK.AMT.  DEP.AMT.   BALANCE

***     01-FEB                             0.00    187.00     187.00
101     01-FEB    K-MART                  10.15      0.00     176.85
102     01-FEB    KROGER                  55.26      0.00     121.59
103     03-FEB    GAS                     21.28      0.00     100.31
***     10-FEB                             0.00   2000.00    2100.31
105     19-FEB    RENT                   500.00      0.00    1600.31
EOF
```

CHECK.BAS printout

BALANCE.BAS

```
10 REM      WRITTEN BY----DON THOMAS
20 REM                   3797 SOUTHFIELD
30 REM                   ST. JOSEPH, MICH. 49085
40 CLEAR
50 CLEAR 4000
60 V=200
70 DIM CC(V),CN(V),DI(V)
80 F$="$$#,###.##"
90 PRINT CHR$(27)+"E"
100 PRINT TAB(5)" ";
110 INPUT "Enter bank's balance for the end of the month ";BB
120 PRINT :PRINT
130 PRINT "List outstanding Deposits."
140 W=W+1
150 INPUT "          Deposit Amount ";DI(W)
```

```
160 DD=DD+D1(W)
170 IF D1(W)=0 THEN 190
180 GOTO 140
190 PRINT :PRINT
200 PRINT "List all outstanding Checks."
210 J=J+1
220 INPUT "Check number ";CN(J)
230 IF CN(J)=0 THEN 270
240 INPUT "             Check amount ";CC(J)
250 B=B+CC(J)
260 GOTO 210
270 F=DD+BB:F=F-B
280 PRINT CHR$(27)+"E"
290 PRINT "Make sure you have entered all service charges and dividends "
300 PRINT "to your checkbook.
310 PRINT
320 PRINT CHR$(27);CHR$(70)
330 PRINT "[illegible]"
340 PRINT CHR$(27);CHR$(71)
350 PRINT "Your end of month balance is ";
360 PRINT USING F$;F
370 PRINT CHR$(27)+"F"
380 PRINT "iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii"
390 PRINT CHR$(27)+"G"
400 PRINT:PRINT
410 PRINT"Bank Bal:";:PRINT USING F$;BB;
420 PRINT"  Outstanding Dep:";:PRINT USING F$;DD;
430 PRINT"  Outstanding Checks:";:PRINT USING F$;B
440 PRINT"----------------   --------------------   ----------------"
"------------"
450 PRINT:PRINT
460 PRINT
470 INPUT "Does this agree with your checkbook balance";A$
480 PRINT
490 A$=LEFT$(A$,1):IF A$="Y" OR A$="y" THEN 600
500 INPUT "What is your checkbook balance";CB
510 GB=F-CB:GB=ABS(GB)
520 PRINT:PRINT"The difference between your checkbook and the statement is";
530 PRINT USING F$;GB;:PRINT " Dollars"
540 PRINT :INPUT "Do you wish to see your outstanding checks listed";PS$
550 PS$=LEFT$(PS$,1):IF PS$="Y" OR PS$="y" THEN GOSUB 640
560 PRINT
570 INPUT "Do you wish to see your outstanding deposits listed";P$
580 P$=LEFT$(P$,1):IF P$="Y" OR P$="y" THEN GOSUB 770
590 PRINT
600 INPUT"Do you wish to try again <N> ";J$
620 J$=LEFT$(J$,1):IF J$="Y" OR J$="y" THEN 10 ELSE SYSTEM
630 PRINT
640 T=J-1:J=1
650 INPUT "Do you wish this printout to be on the printer";FF$
660 FF$=LEFT$(FF$,1):IF FF$="y" OR FF$="Y" THEN OP$="LP:" ELSE OP$="TT:"
670 OPEN "O",#1,OP$
680 PRINT #1,;" "
690 PRINT #1,;"Check #";:PRINT #1,;TAB(20)"Amount"
700 FOR K=1 TO T
710 PRINT #1,; CN(J);,
720 PRINT #1,; USING F$;CC(J)
730 J=J+1
740 NEXT K
750 CLOSE
760 RETURN
770 WW=W:WW=WW-1
780 W=1
790 INPUT "Do you wish to see the deposits on the printer";A$
800 A$=LEFT$(A$,1):IF A$="Y" OR A$="y" THEN LP$="LP:" ELSE LP$="TT:"
810 PRINT
820 OPEN"O",#1,LP$
830 PRINT #1,;"Outstanding Deposits"
840 FOR Z=1 TO WW
850 PRINT #1,;USING F$;D1(W)
860 W=W+1
870 NEXT Z
880 CLOSE
890 RETURN
```

BE A HAPPY HUGGER.

Z-100 Users! Call us for floppy & hard disk add-on prices. We have 8" & 5¼" drives for you!

Get to know Floppy Disk Services.

We are contracted dealers for Siemens, Shugart and Tandon disk drives. You may do a double take when you see our prices, but there's no catch—we simply buy in large quantities and sell at reasonable prices.

We've sold thousands of drives to our Heath/Zenith friends. And with our good service and fair guarantee, we keep our friends friendly.

Check our prices on these Heath/Zenith compatible drives . . .

FDD-100-5 'flippy' (Just like your internal H89 drive) $215.00

FDD-221-5 DS/DD 96tpi (80 track) same as TM-100-4 ..... 330.00

FDD-200-5 same as 100-5 but double sided (2 heads) 250.00

FDD-100-8d5 SS/DD 8 inch with 3ms step! ..... Sale! 240.00
2 for $450.00 ea.

FDD-200-8 DS/DD 8 inch 3ms step ..... Sale! 345.00

New Shugart half height disk drives!!

SA-455 double sided, double density, 48tpi half height 5¼" step time to 6ms! 295.00
2 for $245.00 ea.

SA-465 double sided, double density, 96tpi half height 5¼" step time to 6ms! 350.00
2 for $295.00 ea.

SA-860 double sided, double density, 8 inch half height. Step time to 3ms! 495.00
2 for $445.00 ea.

PAYMENT POLICY We accept Mastercard, Visa, personal checks & M.O. We reserve the right to wait 10 working days for personal checks to clear your bank. All shipping standard UPS rates unless otherwise requested. New Jersey residents *must* add 6% sales tax.

Due to production deadlines, prices in this ad could be as old as 2 months. If in doubt, call!

Prices and specs subject to change without notice.

Need an enclosure for 5¼ or 8"?

Single vertical case w/ps A&T styled to match Heath color ..... $ 50.00

Dual vertical case with power supply A&T 75.00

Dual horizontal case with dual floppy supply A&T 125.00

Dual horizontal 8" A&T 275.00

Half Height Enclosures

Floppy Disk Services has designed these new half height enclosures from the ground up! We had you, the end user in mind and designed these 5 models:

Vertical dual 5¼ half height ..... $ 75.00

Single full size 5¼ or up to 2 half height 125.00*

Dual side by side horizontal half height 5¼ 125.00*

Dual side by side horizontal half height 8 inch ... 295.00*

Single full size 8 inch or up to 2 half height 295.00*

* These enclosures are of professional design and have data cableing options available at extra charge.

** all of the above enclosures include A&T with power supply.

Magnolia 8" Controller

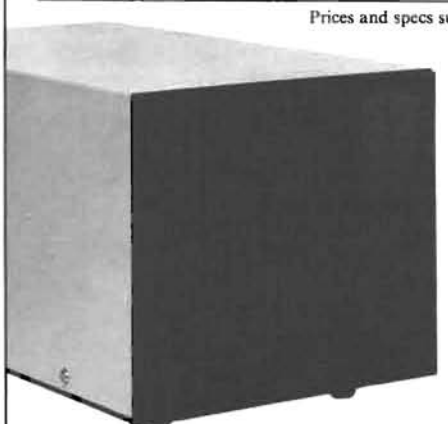
Magnolia Microsystems controller for the Heath H88, 89, 90 enables you to use any combination of 8 or 5¼ single or double sided, single or double density—up to 4 of each size drive! You even get CP/M in 8 and 5¼ format serialized for you. This controller is available at a special price of $525.00.

Our pledge to you . . .

All Floppy Disk products are brand new and 100% warranted. If you are unhappy with our systems or components for any reason, we will refund your money promptly. There are two requirements: 1. the equipment must be in as good shape as you received it; and 2. you must call or write for a return authorization. This offer is good for 30 days, beginning with your invoice date. Feel free to call us with any questions.

We make Huggers happy. Call today!

Toll Free Order Line **(800) 223-0306**
Tech Help or Info. **(609) 799-4440**



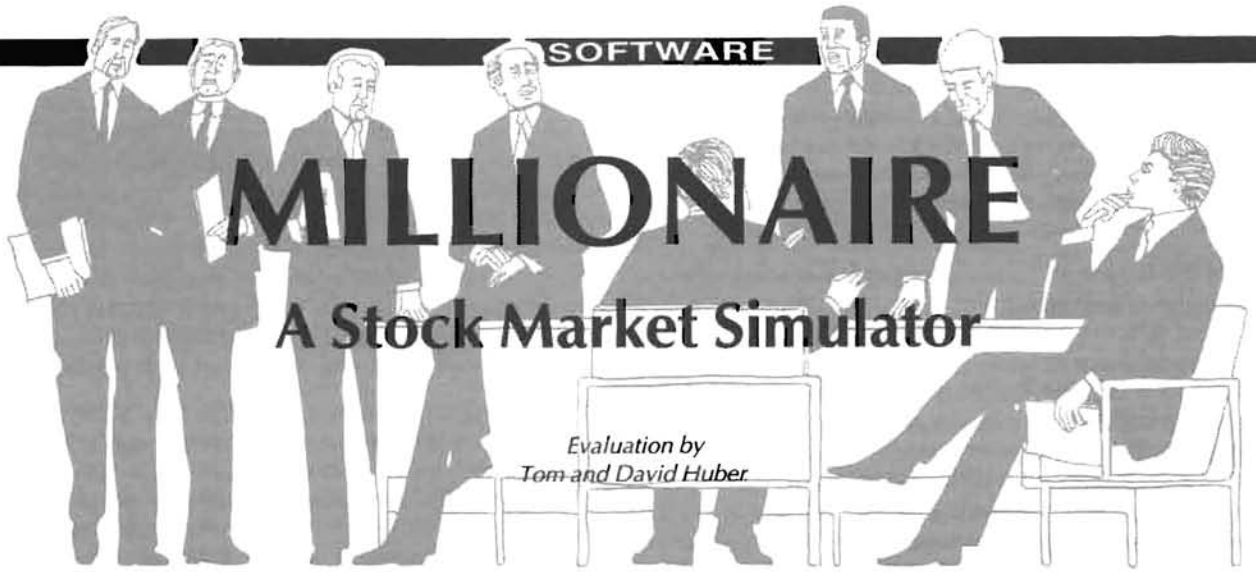
Dual Heath Add-on FDD-100-5

Dual Half Height or Single Full Size 8"

FLOPPY DISK SERVICES™ INC.

741 Alexander Rd.
Princeton, NJ 08540

CP/M and H88-89-90 are registered trademarks of Digital Research and Heath Co., respectively.



MILLIONAIRE

A Stock Market Simulator

Evaluation by
Tom and David Huber

"Millionaire, The Stock Market Simulation," is an offering from Blue Chip Software. I had the opportunity to put this educational stock market simulator through its paces and can readily report that it is also a very challenging game.

In our home, educational simulations have proven to be the most popular and enduring of computer games. Our favorite is George Blank's Santa Paravia en Fiumaccio, first published for the TRS-80 in Softside Magazine a number of years ago. The game is an excellent takeoff of the traditional Hamurabi (originally published in David Ahl's "101 BASIC Computer Games" (DEC) and has provided us with hours of entertainment in both a solitaire and competitive atmosphere. Millionaire has the potential to be just as entertaining and educational.

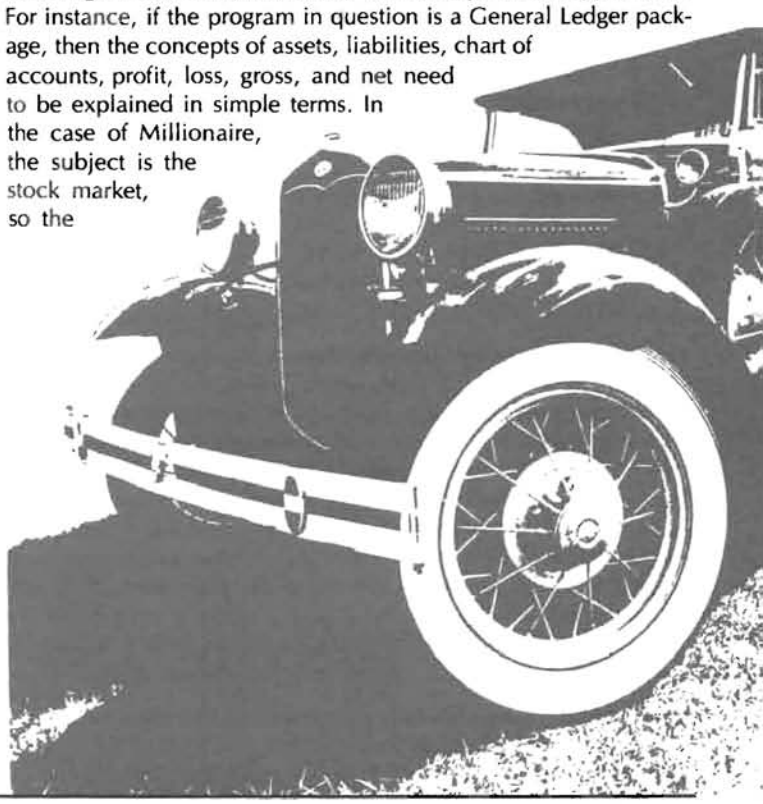
Essentially billed as a "Stock Market Simulation," Millionaire simulates the rise and fall of the stock market (and perhaps your savings as well). Five major industries (Computer, Oil & Gas, Retail, Auto, and Heavy) are represented by three stocks from each group. The represented companies are Control Data, IBM, and NCR (Computers); Conoco, Exxon, and Mobil (Oil and Gas); K-Mart, Sears, and Tandy-Radio Shack (Retail); GM, American Motors, and Bendix (Auto); and United States Steel, Dow Chemical, and Caterpiller Tractor (Heavy).

As the game progresses through each week, it reflects the somewhat typical behavior of the market with its overall trends, peaks and valleys, and maverick stocks (those that seemingly defy the rest of the market—for good **or** bad). The activity for each week is illustrated with a graph showing previous weeks' performances as well as a selected graph of one of the five industries (based on the performance of the three stocks of that group). The other industry's graphs may also be viewed as well as those for individual stocks.

Weekly news bulletins, which effect stock performance, are randomly displayed. By carefully monitoring these bulletins, one can sometimes judge performance of an individual stock. Occasionally, no news is displayed. When that happens, the market tends to rise a bit, fulfilling the old adage, "no news is good news", although I've had a long losing trend continue right through two or three weeks' worth of "no news". Along with a news report, a capsule summary of stock activity is produced, showing the year's high and low, and week's close and change of each stock. Also shown are new highs and lows, the number of stocks that were "losers" and "winners", the gain or loss of the market (in points), and the performance of an average share.

The play of the game is straight-forward: you buy and sell stocks. Of course, knowing **when** to buy or sell can be somewhat tricky. The simulation also supports corporate histories, loans, margin accounts, and call and put options. The basic functions are explained in the documentation. However, not enough background information is provided to let the novice in on the secrets of buying and selling with "calls" and "puts". However, this is also true of the entire package; if you don't know the first thing about the stock market, then you will feel like you have been assigned to deep left field in a kids softball game.

I have found that most documentation that accompanies software falls short. The programmer and/or author evidently knows the material very well; well enough to assume far too much on the part of the average user. Documentation should be complete enough to explain not only the operation of the program, but also the concepts behind the program. By this I do not mean the programming trade secrets, but rather, the concepts of the application. For instance, if the program in question is a General Ledger package, then the concepts of assets, liabilities, chart of accounts, profit, loss, gross, and net need to be explained in simple terms. In the case of Millionaire, the subject is the stock market, so the



concepts of stocks, investing, margin, call and put options, effect of taxes, all could be explained. For instance, when I consulted the family World Book Encyclopedia, I found margin described, but not call and put transactions. Filling in the concepts can make the difference between average documentation and great documentation. Of course, the documentation has to read easily, as well.

Packaging for this product is outstanding. My first impressions of the padded folder, gold labels and embossing, clean layout, and clear instructions, told me that this product was well worth looking at. All criticism of documentation aside, this product does do what it was intended to do; and, by not being as complete as it could be, the game forced us to visit our local library and find out more about the stock market. An added plus to the game is that it is very educational. It closely parallels situations that have recently happened in the market. The news is timely, well within the last ten years—for instance, the oil embargo, the discovery of oil on Alaska's North Shore, market diversity, labor problems, and the recession.

We (my son and I) tested the CP/M version on an H-100 under the CP/M-85 operating system. For those of you who are not familiar with this operating system, it operates almost the same as an H- 8/H-19 combination, H/Z-89, or Z-90 Computer and graphics under CP/M. Therefore, the results we experienced will be essentially the same as with those machines.

Getting the game up and running was not as easy as one-two-three. The instructions were written (evidently) for an earlier version of the game (one that required a copy of MBASIC, Microsoft's BASIC Interpreter), and so the loading instructions were wrong.

Our machine language version required some undocumented doctoring. After we transferred the files to a bootable CP/M-85 disk, we started Millionaire by entering an M. (We discovered a .COM file by that name.) This brought up the introductory screen, but with a lot of strange numbers and letters in front of each line. In addition, nothing that we entered via the keyboard seemed to display correctly. Going back to the directory, I discovered another file by the name of T.COM. Running this file produced a straightforward display of terminal types. Selecting the Heath 19/89 option, I returned the program. Now everything worked perfectly.

Exploring the T.COM program a bit further, I found that most common terminals were represented and, if you are running a home brew terminal with your H-8 and CP/M, you can even enter the special codes needed for the game and create a new terminal type for your own system. Once set into the system, the terminal characteristics (for clear screen and direct cursor addressing) will be stored and used with the main program every time you start it.

Getting the simulation started is time consuming. However, there is a good reason for this: each of the fifteen stocks has 91 weeks' activity (according to the documentation) created randomly each time the program is started with a new session. This insures a truly random play of the market so that no two games are ever the same. It also appears that the activity is not really random in the true sense of the word. Some very clear trends appear in each game, much the same as a series of trends will appear given any particular world- or nation-wide year and events. This tendency toward real life puts this whole program a mark above the average; the programmer(s) did their research very well and the extra time is well worth the added realism to the real world.

However, the time consumed is long enough to make you wonder if your computer died on you. On our machine, it took about four minutes to get a game started. It took only about a minute and a half if a previous session was being loaded.

The object of the game, to become a millionaire, is accomplished by advancing through different levels (or status) during the play of the game. It will take more than one 91-week session to advance from novice through investor, speculator, professional, broker, and finally to millionaire. Each 91-week session took about an hour to complete. In one session, I obtained nearly 180 percent of my initial investment, while in another session, I saw a loss of better than 20 percent. My son did much worse and quit early on several occasions. However, after he had watched me play through a game or two, he took heart and tried again with much better results.

This experience brings to mind one that I had while teaching computer science at a local community college: high school and junior high students have some problems relating to real life situations. In the case of this simulation, the action of the stock market depends upon a wide number of variables, including many that are not of the obvious cause and effect variety. For some reason it takes the experience that age provides to truly grasp the importance of seemingly unimportant details. In that regard, this simulation, because of the relationship between various types of news events and the result on the market, provides a real teaching tool that is rarely found, even in games like Santa Paravia.

The play is excellent, with plenty of random happenings to keep things interesting. However, we did note several irregularities with real world experience, mostly in the form of the news bulletins. For instance, there was a news bulletin reporting dividend earnings for Tandy stock. Tandy is not a dividend stock—it is a growth stock. In several instances, "wildcat" strikes were reported for companies that have never experienced such labor problems. Items of this nature have no bearing, however, on how the game is played; they simply do not reflect the real world.

The histories given for each corporation are accurate, informative, and interesting. However, the capsules are marred by typing errors, mostly limited to poor spacing around punctuation. Given the cost of this game and care put into the rest of the package, the sloppy typing should have been corrected long ago.

A word of warning to the uninitiated. Like all simulations, there are limitations to the realism achievable, and this game should not be construed in any manner as being exactly identical to the real world. It does come the closest of any simulator to real life stock market investing and activity as any I have seen, but it does not and can not possibly take into account all of the many variables that affect the various stocks in the market.

One feature I would like to see (although it would take away from the realism of the simulation) and would really make this game ideal for pure entertainment: competition against one or more other players. For this reason alone, Millionaire probably will never replace Santa Paravia in our home.

Summary ratings:

Packaging and production	**: 10 (top-notch)**
Documentation accuracy	**: 7 (getting started)**
Documentation completeness	**: 5 (not enough background)**
Documentation on disk	**: 9 (sloppy typographical errors)**
Playing interest	**: 8 (non-competitive)**
Playing challenge	**: 10 (excellent)**
Playing time	**: 8 (slightly long)**
Features	**: 10 (unusually complete)**
Cost to performance value:	
As a game	**: 6 (expensive)**
As a simulation	**: 10 (unusually real)**
For education	**: 10 (with additional assignments)**

Vendor: Blue Chip Software
19824 Ventura Blvd., Suite 125
Woodland Hills, CA 91364
(213) 881-8288
Price: $69.95 (8" disk)
$59.95 (5" disk)
Machines: TRS-80, Apple, IBM PC, Osborne One, and CP/M (including all Heath/Zenith Systems that run 8-bit CP/M version 2.2 or higher).

Reviewers:
Tom Huber, originally from the Pacific Northwest, has been affiliated with the computer, business machine, and electronics industry since 1965, and is currently employed by Heath Company as a computer publications writer/editor. His interests include spectator sports (pro basketball and football, auto and unlimited hydroplane racing), computers and their applications, and raising a family of 3 daughters and one son. He serves as New Products editor for REMark and was editor for 80-U.S. Journal, a TRS-80 publication, for a year.

His son David, who provided valuable play information for this review is 13 and active in school sports. He interests include scouting, basketball, football, bowling, computers and electronics.

Getting Started with Assembly Language

(We're Getting There...)

Pat Swayne
Software Engineer

If you have recently completed an introductory study of assembly language such as the Heath Assembly Language course, then this series of articles is for you. In it, I am attempting to bring you from a theoretical knowledge of assembly programming to a practical one. But if you are reading REMark for the first time, I urge you to obtain the last three issues and read the previous installments of this series.

Part V - Using A Printer In CP/M

As I pointed out last month, new assembly programmers sometimes find using a printer in their programs difficult. This is because the operating system (HDOS or CP/M) does not provide as much support for printers as it does for the console (keyboard and screen). I also pointed out that studying someone else's program does not always help, because they do not explain what they are doing. So, for those of you who are reading only the CP/M installments in this series (Shame on you!), I will repeat the second of my rules for assembly language programming.

Rule 2. *Include a generous amount of comments in your program, with the idea in mind that someone else may be trying to learn from your work, even though your purpose in writing it is not to teach.*

Readers of last month's installment will recall that HDOS treats all input/output devices (console, printer, disk, etc.) in about the same way, and that you can "talk" to any of them using the same system routines. CP/M is very different in this area. Each of its 5 logical devices (Console, List (printer), Reader*, Punch*, and Disk) *(The Reader and Punch devices are throw- backs to earlier days of computing, when paper tape was used for program storage. In modern CP/M systems, they are often used for modern communication.)* is supported by its own system routines (BDOS functions). So while the HDOS discussion on printer output serves as an introduction to disk operations, this CP/M discussion of printers will have no bearing on disk use.

A Computer Typewriter (in CP/M)

As with the HDOS segment on printers, the sample program I have chosen for this discussion turns your computer into a simple typewriter, with the ability to correct a line before it is committed to paper. Listing 1 shows the program, which, as usual, is an assembly translation of a BASIC program. The BASIC program in this case is written for CP/M Microsoft BASIC.

After the initial comments, the program has the usual EQUate table that defines system calls and other parameters that will be used in the program. The only new definitions (that were not in the last CP/M example program) are the LSTOUT BDOS function and an external STACK definition. The other functions and definitions were covered in the last CP/M installment.

The program starts in the same way as the

Listing 1

```
;       TYPEIT.ASM
;       THIS PROGRAM IS AN ASSEMBLY LANGUAGE VERSION OF THE
;       FOLLOWING BASIC PROGRAM
;
;       10 PRINT "TYPE LINES AT YOUR CONSOLE.  THEY"
;       20 PRINT "WILL BE PRINTED (ON YOUR PRINTER)"
;       30 PRINT "WHEN YOU HIT RETURN.  TYPE A PERIOD"
;       40 PRINT "AT THE BEGINNING OF A LINE TO STOP."
;       50 LINE INPUT L$
;       60 IF LEFT$(L$,1)="." THEN END
;       70 LPRINT L$
;       80 GOTO 50
;
;       THIS VERSION IS FOR CP/M, AND USES SOME
;       BUILT-IN CP/M ROUTINES
;
;       BY P. SWAYNE, HUG  4-MAY-83

;       DEFINE CP/M ROUTINES, ETC.

BOTTOM  EQU     0               ;BOTTOM OF USER MEMORY
BDOS    EQU     BOTTOM+5        ;JUMP INTO CP/M BDOS
TPA     EQU     BOTTOM+100H     ;TRANSIENT PROGRAM AREA
CONOUT  EQU     2               ;CONSOLE OUTPUT FUNCTION
LSTOUT  EQU     5               ;LIST DEVICE (PRINTER) OUTPUT
TYPE    EQU     9               ;BLOCK TEXT TYPE FUNCTION
LINPUT  EQU     10              ;LINE INPUT FUNCTION
STACK   EQU     100H            ;PUT STACK BELOW TPA

        ORG     TPA

;       SET UP STACK

START   LXI     H,0             ;ZERO HL REGISTER
        DAD     SP              ;ADD STACK POINTER VALUE
        LXI     SP,STACK        ;SET OUR STACK POINTER
        PUSH    H               ;SAVE CP/M'S STACK ON OURS

;       10 PRINT "TYPE LINES ... ETC.

        LXI     D,TYPEL         ;POINT TO "TYPE LINES" STRING
        MVI     C,TYPE          ;GET TYPE FUNCTION
        CALL    BDOS            ;TYPE THE MESSAGE

;       50 LINE INPUT L$
```

```
LOOP    LXI     D,LINE          ;POINT TO LINE BUFFER
        MVI     C,LINPUT        ;GET LINE INPUT FUNCTION
        CALL    BDOS            ;INPUT THE LINE
        LXI     H,LINE+1        ;POINT TO NO. OF CHARS TYPED
        MOV     E,M             ;GET THE COUNT
        MVI     D,0             ;DE = COUNT
        INX     H               ;MOVE TO THE START OF THE LINE
        PUSH    H               ;SAVE THIS ADDRESS
        DAD     D               ;ADD CHARACTER COUNT
        MVI     M,0DH           ;INSERT A CR
        INX     H               ;MOVE TO NEXT LOCATION
        MVI     M,8AH           ;INSERT LINE FEED + 80H
        MVI     E,0AH           ;GET A LINE FEED
        MVI     C,CONOUT        ;GET CONSOLE OUTPUT FUNCTION
        CALL    BDOS            ;PRINT LINE FEED ON CONSOLE

;       60 IF LEFT$(L$,1)="." THEN END

        POP     H               ;GET LINE START ADDRESS
        MOV     A,M             ;GET FIRST CHARACTER
        CPI     '.'             ;IS IT A PERIOD?
        JNZ     PRINTL          ;IF NOT, PRINT THE LINE
        POP     H               ;ELSE, RESTORE CP/M'S STACK
        SPHL                    ;SET IT
        RET                     ;RETURN TO CP/M

;       70 LPRINT L$

PRINTL  ANI     7FH             ;STRIP 8TH BIT FROM CHARACTER
        MVI     C,LSTOUT        ;GET LIST OUTPUT FUNCTION
        PUSH    PSW             ;SAVE CHARACTER
        PUSH    H               ;SAVE POINTER
        MOV     E,A             ;PUT CHARACTER IN E
        CALL    BDOS            ;PRINT THE CHARACTER
        POP     H               ;RESTORE ADDRESS
        POP     PSW             ;RESTORE CHARACTER
        CMP     M               ;COMPARE WITH ORIGINAL CHARACTER
        INX     H               ;MOVE TO NEXT LOCATION
        MOV     A,M             ;GET NEXT CHARACTER
        JZ      PRINTL          ;END NOT FOUND, CONTINUE

;       80 GOTO 50

        JMP     LOOP            ;LOOP UNTIL USER QUITS

;       DATA AND STORAGE

TYPEL   DB      'TYPE LINES AT YOUR CONSOLE.  THEY',13,10
        DB      'WILL BE PRINTED (ON YOUR PRINTER)',13,10
        DB      'WHEN YOU HIT RETURN.  TYPE A PERIOD',13,10
        DB      'AT THE BEGINNING OF A LINE TO STOP.',13,10,'$'

LINE    DB      80,0            ;LINE INPUT BUFFER

        END     START
```

CP/M console example, by getting the value of the stack pointer as maintained by CP/M, setting a new stack for the program, and saving the CP/M stack value. Before, however, we used some space within the program for the stack, but this time we are going to use an external stack. The label STACK is made equal to 100H, which means that the stack will start at the Transient Program Area (TPA) where the program starts and work down. This practice is acceptable as long as your program will not be using that area of memory, or will not use any BDOS functions that use that area, and you do not use up more than 128 bytes of stack space. The area just below the TPA is called the "default DMA area", and is often used for disk operations. Since our program performs no disk I/O, we can use that area for a stack and save space within the program itself.

The first four lines of the BASIC program are translated using the BDOS Type function. Line 50 is translated by the BDOS Line Input function, and is done like the previous CP/M program with two notable differences. The first is that there is no reserved space within the body of the program for the input buffer, but instead it is just put at the end of the program, and characters typed into it just extend beyond the program. This practice is acceptable as long as the buffer area required is of a definite size that is not so big as to overwrite the operating system. In this case, only 82 bytes maximum beyond the end of the program are required.

The second difference in this program is that the input string is terminated with a line feed with the high bit set (line feed plus 80H), instead of with a dollar sign as before. This is because we are not going to print the input line on the console using CP/M's text string printing function, but on a printer one character at a time. Unlike HDOS, CP/M cannot print blocks of text in one operation, so a programmer must develop his/her own routine to print the text, and therefore can use anything he/she wants to terminate the block of text.

The line input section of the program saves the address of the start of the input line of text so that the first character can be checked to see if it is a period (to translate line 60 of the BASIC program). If it is, the program returns control to CP/M by restoring the CP/M stack and RETurning to the Console Command Processor.

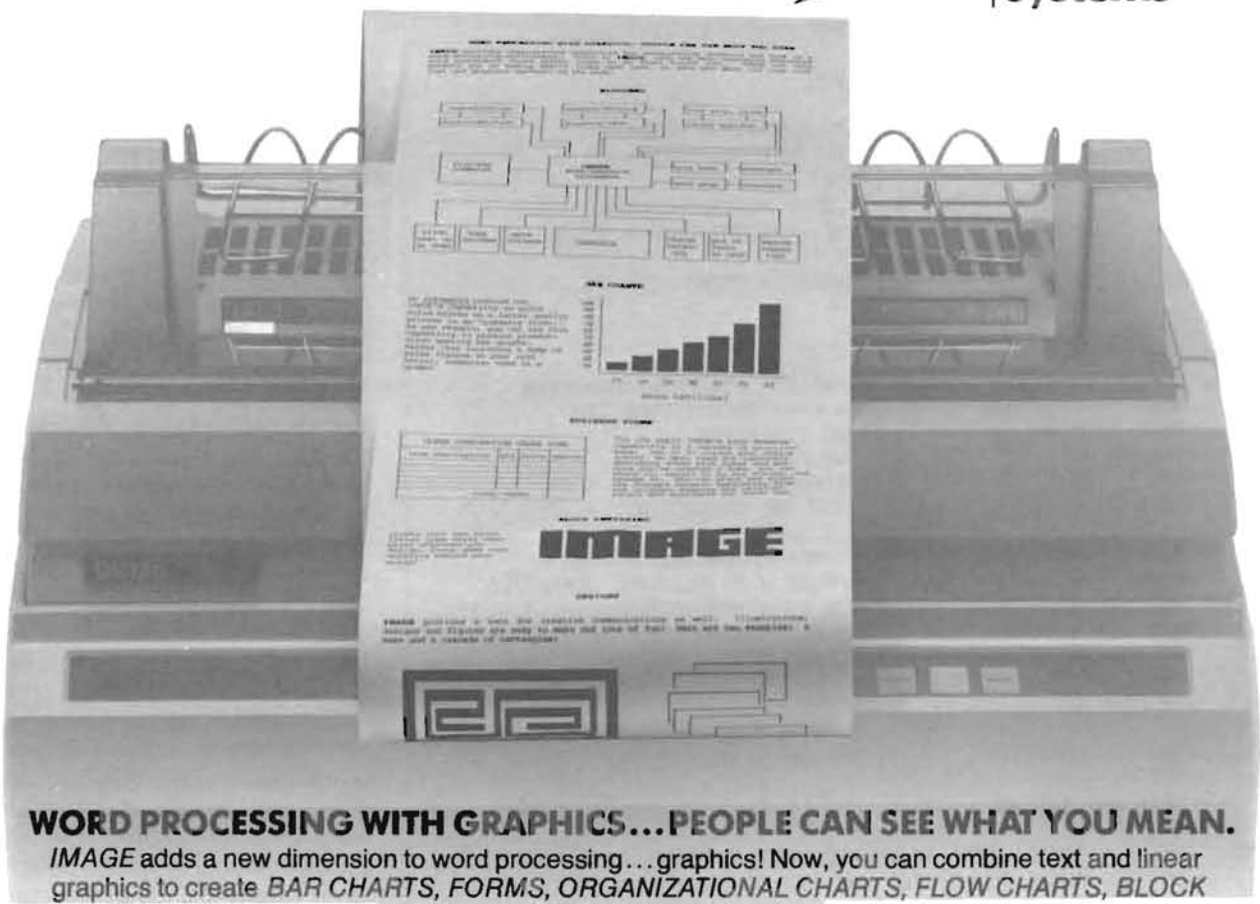
We arrive at the section of the program that translates line 70 with the HL register already pointing to the beginning of the line of text and the first character in the A register. The first thing that is done is to AND the character with 7FH to strip the 8th bit, in case it is set. Then the character is printed using CP/M's one and only BDOS function call (function 5) that directly supports the List (printer) device. While this is done, the A, F, and HL registers are saved on the stack because CP/M does not guarantee preservation of any registers during BDOS function calls. After the registers are restored, the program compares the stripped character in the A register with the one in memory, which should be the same unless it is the last character in the text string. The text pointer is incremented and the next character is taken from memory, and then the program jumps back to print another character unless the end of the text was found by the compare operation.

Line 80 is translated by a jump back to the label LOOP. This is followed by the data and storage area that holds the introductory screen message (at the label TYPEL) and the start of the line input buffer (at LINE). The number of characters that can be typed into one line is limited to 80 here, and the actual text area extends beyond the program end.

Next month, I plan to depart from the procedure I have been following and examine someone else's program, to see where rules are followed or broken. Then in the following months, we will get into disk operations.

WORD PROCESSING IS NOW MORE THAN PROCESSING WORDS.

ANNOUNCING IMAGE™ FOR ZENITH data systems



WORD PROCESSING WITH GRAPHICS...PEOPLE CAN SEE WHAT YOU MEAN.

IMAGE adds a new dimension to word processing... graphics! Now, you can combine text and linear graphics to create *BAR CHARTS, FORMS, ORGANIZATIONAL CHARTS, FLOW CHARTS, BLOCK LETTERS,* and much more. The result? More powerful and more effective communication. People can *SEE* what you mean.

TOP MARKS FROM INFOWORLD

IMAGE won top marks from *INFOWORLD* for its innovation, quality, reliability, and ease of use:

"*IMAGE* certainly deserves accolades in the *performance* category."

"The *documentation* is simply superb. It is professionally done from cover to cover."

"Without a doubt, this program is *easy to use*."

"The program is *bombproofed* so well that I had trouble finding any errors."

IMAGE is a trademark of MicroArt Corporation.

InfoWorld
Software Report Card

Image

	Poor	Fair	Good	Excellent
Performance	□	□	□	■
Documentation	□	□	□	■
Ease of Use	□	□	□	■
Error Handling	□	□	□	■

All quotes are from InfoWorld's *IMAGE* software review, by Marty Petersen, June 14, 1982.

EXTRAORDINARY VALUE...

"The modest $295 Price Tag is a Bargain."

IMAGE runs on any Zenith Z/89 or Z/90 computer, on any Heath H/89 or H/90 computer, or on any Z-80-based CP/M system linked to a Z/19 or H/19 terminal.

TO ORDER CALL TOLL FREE:
1-800-MICROART (1-800-642-7627)
in Oregon, call: **1-692-3950**

OR WRITE:
MICRO ART
MicroArt Corporation
200 Market Bldg.
Suite 961
Portland, OR 97201

Mastercard, Visa and COD orders accepted.

A Simple Separate Keyboard For The H/Z-89

Ernst S. Duesterhoeft
Box 37
Helenville, WI 53137

There have been many times that I longed for the convenience of a separate keyboard on my H/Z-89.

While there are many places where the convenience of the all-in-one makes it desirable, using the H/Z-89 for word processing on a desktop is not one of them. I was continually searching for the best arrangement, but nothing seemed to work. This went on for quite some time until I saw a TRS-80 Model II with its separate keyboard. I knew that I had found the solution.

The first thing that I did was to turn my H/Z-89 (H/Z-19) on its side and remove the six screws which hold the keyboard in place, lifted it out and disconnected the flat cable which plugged into the PC board under the keyboard.

With the keyboard laying on the bench I examined the construction of the base for the '89'. I found that there would be no structural problems involved in removing the front part of the base. When I measured the piece of the base which would remain, I found that it would be about an inch short of being deep enough to hold the keyboard. After thinking about it for a while I decided that it would be a simple matter to fabricate an extension to the rear of the base out of several small pieces of wood. Using 5-minute epoxy, these could be glued into place and with a small amount of sanding would give a finished look to the case for the keyboard. A similiar solution was used to cover the hole under the frame for the CRT on the base of the H/Z-89. These modifications can all be accomplished without the use of power tools.

The only part of the modification that held any doubts in my mind was connecting the keyboard to the H/Z-89. Unlike the TRS-80, which converts the keyboard information to a serial format and uses a shielded 5 wire cable, the H/Z-89 uses a 34 conductor flat cable to connect the keyboard to the terminal board. The possibility of stray noise getting into the keyboard controller and causing errors was greatly increased with an unshielded cable lying around unprotected. This fear has proven to be unfounded, since in several weeks of use not a single error has been detected. Also I have not been able to detect a change in the level of radio-frequency interference as a result of this modification.

The two rubber feet which were fastened to the front part of the base should be removed and replaced under the main part of the computer. This is easily accomplished by removing the outside two of the four screws which hold the front panel in place, discarding the flat washers, and installing the rubber feet.

The only parts which I needed to purchase were four small rubber feet to place under the keyboard and a 36 inch, 34 cond. flat cable assembly with a male connector on one end and female connector on the other.

To use the keyboard, it is a simple matter of connecting it to the computer with the extension cable. The only caution required is to make sure that the correct position of pin one must be maintained.

An alternate solution to the cable would be to use a cable with a female socket on each end and connect directly to the terminal PC board and then route the cable out the rear of and underneath the cabinet.

To finish the project I plan to custom mix both colors used on the H/Z-89. I will use a small spray gun to apply the finish. If there is sufficient interest I can have small spray cans filled. For further information please send inquiries to me, Ernst S. Duesterhoeft, Box 37, Helenville, WI 53137.

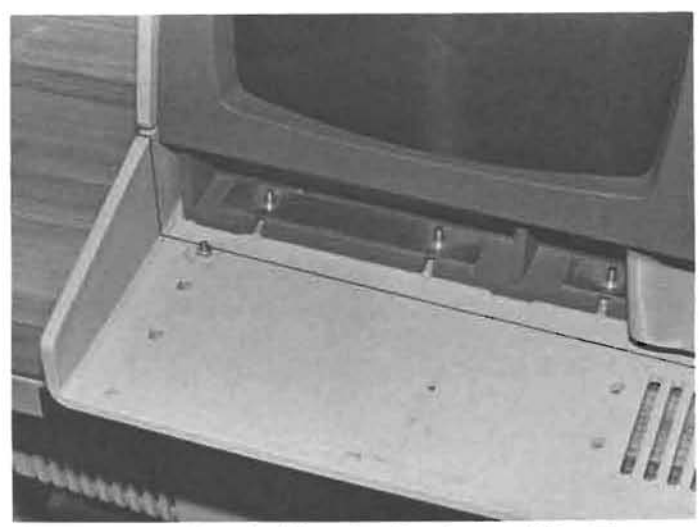


Photo 1. Front view of cabinet showing the cut-off line. The screw which is shown under the cut-off line holds a rubber foot which should be moved to the frame screw immediately above it.

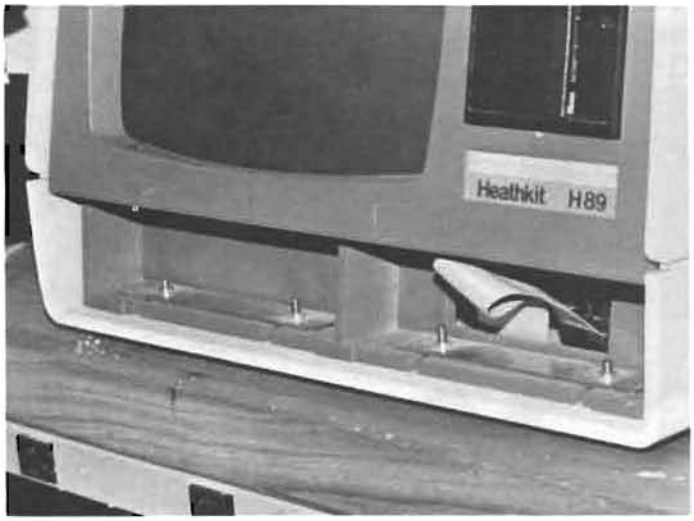


Photo 2. Front view of cabinet after keyboard base has been cut off. At this point the two front rubber feet have been relocated under the two outside frame screws.

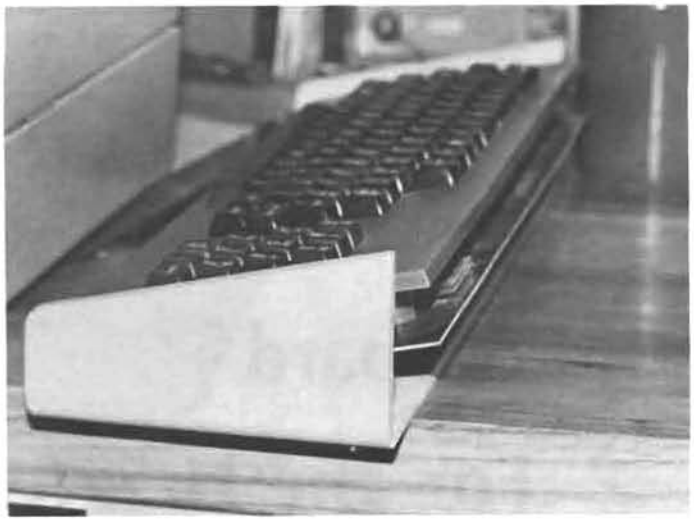


Photo 3. Side view of keyboard after cutting off the base.

Photo 4. End view of keyboard base showing the first piece of the extension being clamped after glueing.

Photo 5. View of the keyboard base before final trimming. The three pieces of wood were installed one at a time allowing time for the glue to cure.

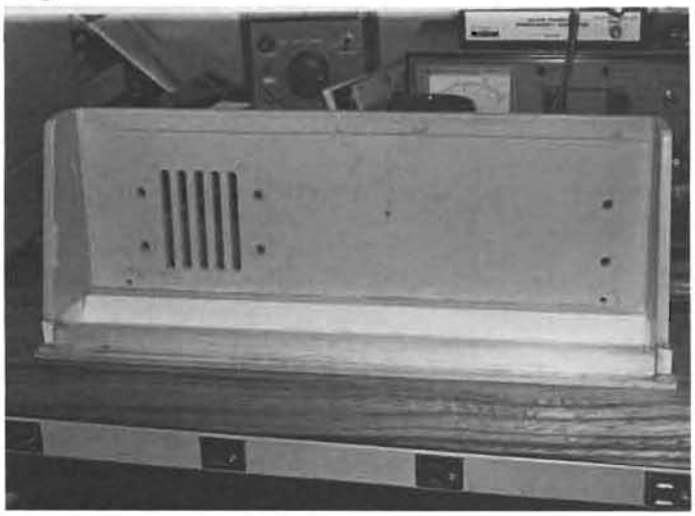
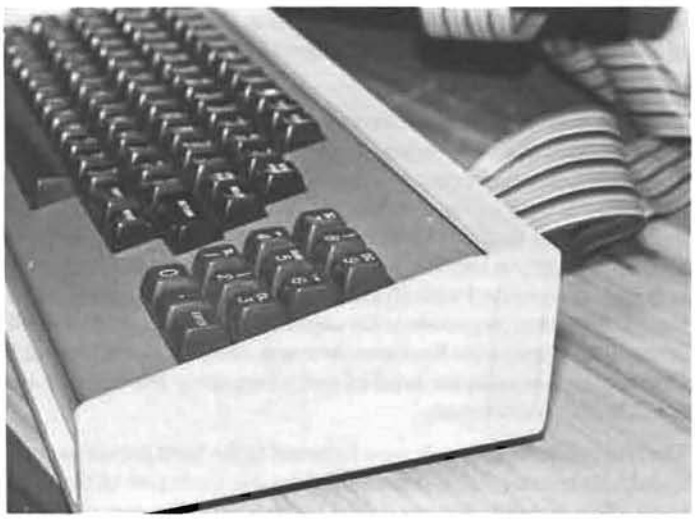


Photo 6. End view of the keyboard base showing the extension to the base after final trimming and sanding. Ready to finish.



"Changing Gears" in Your 4MHz H-89

Peter Shkabara
Analytical Products
29924 Road 168
Visalia, CA 93291

We all owe Pat Swayne a bit of gratitude for publishing his excellent 4MHz conversion article in the November issue of REMark. In using the conversion, however, I soon discovered the need to be able to switch back and forth between the two clock speeds. Since I run under CP/M, it is easy to switch the clock by patching the Control byte. However, this will kill the ability to use disk access! In order to retain disk operations, the BIOS needs to be patched each time the clock speed is changed. SET.COM is a program which I created to perform this task - easily and conveniently.

Several versions of SET.COM have been developed. Some will automatically detect and adjust for the Heath BIOS version and memory size. There is even a version for CDR Systems, Version 2.7 BIOS. The assembly source listing shown here is a simplified version of SET.COM which will still do the required task. This version is easy to change, and will allow itself to be adapted to user modified BIOS or versions not already included in the program.

Only 8080 codes and mnemonics have been used to allow assembly of the file using the ASM.COM assembler which is included with CP/M. To use this listing, the required EQUATES have to be set as needed. Address offsets are already included for Heath CP/M BIOS hard sector Versions 2.2.02 and 2.2.03. For other versions, the user needs to find actual locations of the timing values, calculate their offset in relation to BIOS start address, and then install them at the appropriate locations in the CUSTOM area of the source listing. The CUSTOM EQU equate must be set to TRUE.

For those not familiar with ASM.COM, the procedure to assemble the file consists of calling the utility by typing ASM followed by the name of the assembly source file. For example:

A>ASM SET<CR>

Then sit back and let the program do the work. Don't you wish that you had the 4MHz conversion already installed? Assembly takes half the time at 4MHz. Be sure that the source file is of the type ASM (e.g., SET.ASM).

The assembler will display any errors it finds in the assembly source file. Since the error messages are somewhat cryptic, look carefully at what is indicated to be an error to make any needed corrections. Hopefully you will not have any error messages.

When the assembly procedure is done you will have a file called SET.HEX on the disk. Before the program can be run, it must be converted to a COM file. This conversion is done by the use of the LOAD.COM program which is also a part of the CP/M package. Once you have a HEX file, simply type in LOAD followed by the name of the file to be loaded. For example:

A>LOAD SET<CR>

The loading process is fairly quick, and in almost no time you will have a ready to run program.

For those less adventurous, Analytical Products has been selling a disk with several versions of SET.COM and source code. Also included on the disk are several SUBMIT files with instructions for their use. The SUBMIT files allow the user to automatically create a bootable disk which will boot up at 4MHz. Heath CP/M BIOS Versions 2.2.02 and 2.2.03 and CDR Systems BIOS Version 2.7 are fully supported. Cost of the disk is only $8.00 (California residents add 6% tax) and are supplied in Heath hard sector format.

HDOS and soft sector format versions of SET.COM are under development.

```
;************** SET HEATH BIOS - REMark VERSION ************************
;
;ASSEMBLY SOURCE CODE FOR SET.COM
;
;Created by          Peter Shkabara                          11/27/82
;                    ANALYTICAL PRODUCTS
;                    29924 Road 168
;                    Visalia, CA 93291
;                    209/747-3235
;
; All rights reserved
;
;This program will set the modified H89 to operate at 2 or 4 MHz
;REF REMark Issue 34-November '82 p25
;
;Mod 2/5/83 - adapted for REMark article
;   added IF statements to assemble for version 2.2.02 or 2.2.03
;   and automatic memory size adjustment
;
;4 MHz operation is enabled by setting the '4' bit at output port F2
;
;       Modified by P. Swayne, HUG (pin 9 option and other improvements)
;
;*******************************************************************************
;
; BEGIN EQUATES
;
CBYTE   EQU     00DH            ;LOCATION OF CONTROL CP/M BYTE
;
TRUE    EQU     -1
FALSE   EQU     NOT TRUE
;
;       BELOW, SELECT WHICH PIN OF U552 YOU ARE USING FOR SPEED SWITCH
;
PIN6    EQU     TRUE            ;USE PIN 6 (SET TO FALSE IF PIN 9 USED)
        IF      PIN6
BIT     EQU     4               ;BIT TO SWITCH SPEEDS
        ENDIF
        IF      NOT PIN6                                              ☞
```

```
BIT     EQU     8
        ENDIF
;
; ADDRESS OFFSETS - Replace with your own offset values for
;                   non-standard or other versions of BIOS
;
V02     EQU     FALSE           ;SET TO TRUE FOR VERSION 2.2.02
V03     EQU     NOT V02
CUSTOM  EQU     FALSE           ;SET TO TRUE FOR CUSTOM BIOS
;
; VERSION 02
        IF      V02 AND NOT CUSTOM
WHDA    EQU     0089DH          ;WAIT HOLE DETECT
WHNA    EQU     089AH
WSCA    EQU     08C9H           ;WAIT SYNC CHARACTER
WRITA   EQU     0695H           ;WRITE SECTOR
WRITC   EQU     069DH
READA   EQU     08BDH           ;WAIT SYNC CHARACTER
        ENDIF
;
; VERSION 03
        IF      V03 AND NOT CUSTOM
WHDA    EQU     079DH           ;WAIT HOLE DETECT
WHNA    EQU     07A8H
WSCA    EQU     07C9H           ;WAIT SYNC CHARACTER
WRITA   EQU     0595H           ;WRITE SECTOR
WRITC   EQU     059DH
READA   EQU     07BDH           ;WAIT SYNC CHARACTER
        ENDIF
;
; CUSTOM BIOS  -  ENTER YOUR OWN OFFSET ADDRESS VALUES
        IF      CUSTOM
WHDA    EQU     0H              ;WAIT HOLE DETECT
WHNA    EQU     0H
WSCA    EQU     0H              ;WAIT SYNC CHARACTER
WRITA   EQU     0H              ;WRITE SECTOR
WRITC   EQU     0H
READA   EQU     0H              ;WAIT SYNC CHARACTER
        ENDIF
;
        ORG     100H
;
        LDA     0002H           ;GET BIOS HIGH ADDRESS
        STA     MEMSZ           ;SAVE IT
        LDA     CBYTE           ;GET CONTROL BYTE
        ANI     BIT             ;CHECK 4 BIT
        RRC                     ;DIVIDE BY 2
        IF      NOT PIN6
        RRC                     ;DIVIDE BY 4 IF PIN 9
        ENDIF
        ADI     32H             ;CONVERT TO ASCII
        STA     NOW             ;SAVE PRESENT SPEED
        LXI     D,LOGO
        MVI     C,9
        CALL    5               ;PRINT SIGNON MESSAGE
GET     MVI     C,1
        CALL    5               ;GET CONSOLE INPUT CHARACTER
        CPI     1BH
        RZ
        STA     SPEED           ;PUT SPEED IN MSG
        CPI     '4'             ;TEST FOR '4' INPUT
        JNZ     MEG2            ;IF NOT=4 JUMP
;
; 4MHz ROUTINE
;
        LXI     B,WHDA4         ;GET 4 MHZ VALUES
        LDA     CBYTE           ;GET CONTROL BYTE
        ORI     BIT             ;SET 4MHZ BIT
INSVALS STA     CBYTE
        LDA     MEMSZ           ;GET BIOS START ADDRESS
        MOV     D,A
        MVI     E,0             ;DE = BIOS START
        LXI     H,WHDA          ;GET WHDA OFFSET
        DAD     D               ;ADD BIOS START
        LDAX    B               ;GET NEW VALUE
        INX     B               ;MOVE TO NEXT
        MOV     M,A             ;HOLE DEBOUNCE
        LXI     H,WHNA
        DAD     D
        MOV     M,A             ;HOLE DEBOUNCE
        LXI     H,WSCA
        DAD     D
        LDAX    B               ;GET NEW VALUE
        INX     B               ;MOVE TO NEXT
        MOV     M,A             ;25 CHAR LOOP COUNT
        LXI     H,WRITA
        DAD     D
        LDAX    B               ;GET NEW VALUE
        INX     B               ;MOVE TO NEXT
        MOV     M,A             ;GUARDBAND COUNT FOR WRITE
        LXI     H,WRITC
        DAD     D
        MOV     M,A             ;TWO CHAR DLY BEFORE WRITE
        LXI     H,READA
        DAD     D
        LDAX    B               ;GET NEW VALUE
        MOV     M,A             ;DELAY BEFORE HUNT MODE
LOGOF   LXI     D,MSGOF
        MVI     C,9
        CALL    5               ;PRINT END MESSAGE
        RET                     ;END OF PROGRAM
;
; 2MHz ROUTINE
;
MEG2    CPI     '2'
        JNZ     ERROR
        LDA     CBYTE
        ANI     -1-BIT AND 0FFH ;CLEAR 4MHZ BIT
        LXI     B,WHDA2         ;GET 2 MHZ VALUES
        JMP     INSVALS         ;INSERT THEM
;
MSGOF   DB      08H,'SET TO '
SPEED   DB      '2 MHz'
        DB      0DH,0AH,'$'
;
ERROR   LXI     D,ERMSG
        MVI     C,9
        CALL    5
        JMP     GET
;
; VALUES FOR 2 and 4 MHz
;
WHDA2   DB      014H    ;HOLE DEBOUNCE
WSCA2   DB      050H    ;25 CHAR LOOP COUNT
WRITA2  DB      014H    ;WRITE COUNT GUARDBAND
READA2  DB      030H    ;DELAY BEFORE HUNT
WHDA4   DB      02EH
WSCA4   DB      0A0H
WRITA4  DB      02CH
READA4  DB      064H
;
MEMSZ   EQU     $
        DB      00H             ;LOCATION TO SAVE BIOS HIGH ADDRESS
;
ERMSG   DB      07H,08H,1BH
        DB      'M$'            ;BEEP AND CLEAR CHARACTER
;
LOGO    DB      1BH,'E' ;CLEAR SCREEN
        DB      0DH,0AH,0AH
        DB      'SET PROGRAM - Created 11/27/82 by Peter Shkabara'
        DB      0DH,0AH
        DB      'Version 1.01 12/22/82  HEATH V2.2.0'
        IF      V02
        DB      '2'
        ENDIF
        IF      V03
        DB      '3'
        ENDIF
        DB      0DH,0AH
```

```
        DB      'Sets 2MHz or 4MHz operation by changing Control BYTE'
        DB      0DH,0AH
        DB      'and modifies HEATH BIOS (in memory only)'
        DB      0DH,0AH,0AH
        DB      'Present operation at '
NOW     DB      '2 MHz'
        DB      0DH,0AH,0AH
        DB      09H,'[2] = 2 MHz'
        DB      0DH,0AH,0AH
        DB      09H,'[4] = 4 MHz'
        DB      0DH,0AH,0AH
        DB      'Enter selection / <ESC> to quit'
        DB      0DH,0AH
        DB      '$'
;
        END
```

Using AUTOEXEC.BAT

To Automatically Load A Z-BASIC Program

Gerry Kabelman, C.E.T.
Zenith Data Systems

The AUTOEXEC.BAT file in Z-DOS is a series of ASCII commands to automatically do something when Z-DOS is booted-up. The AUTOEXEC.BAT is called a batch file. Several batch files may be linked together. First let's take a look at a simple one command batch file and then some more complex files.

Before writing a batch file, it is a good idea to sit down and decide what is going to be done. For example, if you wish to load Z-BASIC and run the program DEMO.BAS, the following line will have to be executed.

ZBASIC DEMO

The above line is the only thing that needs to be in the batch file. The batch AUTOEXEC.BAT is the file batch looked for by Z- DOS on boot-up. If a file with the name AUTOEXEC.BAT is found, then Z-DOS will attempt to execute that file. If the AUTOEXEC.BAT is NOT found, Z-DOS will ask for the date and time and place the operator at the Z-DOS command prompt.

What is the easiest way to create a batch file? Well, since my background is in Z-BASIC, I found that a Z-BASIC program will do the job very nicely.

```
10 '       AUTO.BAS  Version 05.25.83  GK:
20 OPEN"O",1,"AUTOEXEC.BAT"
30 PRINT #1,"ZBASIC DEMO"
40 CLOSE:END
```

That is one of the simplest programs around, however, it is very useful in creating batch files for Z-DOS. For HDOS users, do you remember the PROLOGUE.SYS? (See Issue #9, page 14 of REMark for a sample listing of PROLOGUE.ASM.)

Using a batch file allows Z-DOS to by-pass the DATE and TIME commands for programs that do not require the use of the DATE or TIME.

The above program is about as simple a batch file that there can be, however, many additional commands could be added. For example, the time and date for those programs requiring them.

23 PRINT #1,"DATE"
24 PRINT #1,"TIME"

Another addition could be the DATETIME feature that Frank Clark wrote for Issue 40, Page 31 of REMark.

22 PRINT #1,"DATETIME"
25 PRINT #1,"DATETIME"

Other commands such as the directory command (DIR), copying disk (DSKCOPY) or any other command under Z-DOS may be used in a batch file.

Linking several batch files may be done by providing the last line of a batch file as the link to the next batch file. Using the above example, another line could be added to link to the AUTO2.BAT batch file.

35 PRINT #1,"AUTO2.BAT"

The AUTO2.BAT batch file could be created like this.

```
10 '       AUTO2.BAT   Version 05.25.83   GK:
20 OPEN"O",2,"AUTO2.BAT"
30 PRINT #2,"DIR"
40 PRINT #2,"ZBASIC DEMO2"
50 PRINT #2,"AUTOEXEC.BAT"
60 CLOSE:END
```

The AUTO2.BAT will first take a look at the directory of the default drive, load Z-BASIC, and the program DEMO2.BAS. Once the DEMO2.BAS program is finished and the SYSTEM command is executed by either Z-BASIC or the operator, the AUTOEXEC.BAT file is again run, leaving us in a continuous loop. A CTRL-C during the execution of the batch file will prompt us to end the batch file.

Two special commands have been set aside for use in batch files, they are REM and PAUSE. The REM command may be used like the REM command in Z-BASIC, for adding comments to assist the user. When using the REM command, the information after the REM will be printed on the screen to tell the user that maybe a program is loading or to please stand by, etc.

32 PRINT #2,"REM This is the directory of the default drive."

The PAUSE command will print any message following the PAUSE and ask that any key be pressed to continue. A CTRL-C may be pressed at this time to terminate the batch command.

34 PRINT #2,"PAUSE Press CTRL-C to terminate or"

Try using batch files, as they will turn your Z-100 into a turn-key type system.

Addressing Envelopes/Labels

Charles 'Karl' J. Romer
P.O. Box 8796 CRB
Tucson, AZ 85738

```
10 CL$=CHR$(27)+"E": ' ADDRINL.BAS for CP/M 2.203 and MX-80 FT (04 Dec 82)
20 PRINT CL$ TAB(8)"* * * ADDRESSING ENVELOPES/LABELS - INLINE * * *":PRINT
30 PRINT"ENTER PERSON'S NAME - if not a person's name PRESS <CR>:-"
40 LINE INPUT N$:IF N$=""THEN PRINT"ENTER COMPANY NAME:";:LINE INPUT C$:GOTO 70
50 PRINT:PRINT"ENTER PERSON'S TITLE - if no title PRESS <CR>:-";:LINE INPUT T$
60 PRINT:PRINT"ENTER COMPANY NAME-if no company name PRESS <CR>:";:LINE INPUT C$
70 PRINT:PRINT"ENTER ADDRESS - if no address PRESS <CR>:-";:LINE INPUT A$
80 PRINT:PRINT"ENTER CITY, STATE and ZIP:-";:LINE INPUT L$:PRINT CL$
90 PRINT"PRESS 'R' FOR REVIEW OR 'P' FOR PRINTOUT:- ";:RP$=INPUT$(1):PRINT RP$
100 PRINT:IF RP$="R"OR RP$="r"OR RP$="P"OR RP$="p"THEN 110 ELSE 90
110 EM$=CHR$(27)+"E":DM$=CHR$(27)+"F"
120 IF RP$="R"OR RP$="r"THEN EM$="":DM$="":POKE 3,105
130 PRINT"PRESS 'E' FOR ENVELOPE OR 'L' - LABEL(s):- ";:EL$=INPUT$(1):PRINT EL$
140 PRINT:IF EL$="E"OR EL$="e"THEN 170 ELSE IF EL$="L"OR EL$="l"THEN 160
150 GOTO 130
160 INPUT"ENTER NUMBER OF LABELS DESIRED - PRESS <CR>:- ",NL:PRINT
170 IF RP$="R"OR RP$="r"THEN 190
180 PRINT"PREPARE PRINTER - WHEN READY PRESS <CR>:-";:LINE INPUT Z9$:POKE 3,169
190 A=LEN(N$):B=LEN(T$):C=LEN(C$):D=LEN(A$):E=LEN(L$):IF B>A THEN A=B
200 IF C>A THEN A=C
210 IF D>A THEN A=D
220 IF E>A THEN A=E
230 IF EL$="E"OR EL$="e"THEN M=35 ELSE M=INT(16-A/2)
240 IF EL$="L"OR EL$="l"THEN FOR I=0 TO NL:S=0:IF I=NL THEN 540
250 IF N$=""THEN IF T$=""THEN IF A$=""THEN LPRINT TAB(M)EM$C$:GOTO 270
260 GOTO 280
270 LPRINT:S=S+2:GOTO 320
280 IF N$=""THEN LPRINT TAB(M)EM$C$:S=S+1:GOTO 310
290 LPRINT TAB(M)EM$N$:S=S+1:IF T$=""THEN 300 ELSE LPRINT TAB(M)DM$T$;S=S+1
300 IF C$=""THEN 310 ELSE LPRINT TAB(M)DM$C$;S=S+1
310 IF A$=""THEN 320 ELSE LPRINT TAB(M)DM$A$;S=S+1
320 LPRINT TAB(M)DM$L$:IF EL$="L"OR EL$="l"THEN 340
330 IF RP$="R"OR RP$="r"THEN 380 ELSE 540
340 IF S=2 THEN LPRINT:LPRINT:LPRINT:GOTO 370
350 IF S=3 THEN LPRINT:LPRINT:GOTO 370
360 IF S=4 THEN LPRINT:GOTO 370
370 NEXT
380 PRINT:PRINT"NEED TO MAKE CORRECTIONS? (Y/N):- ";:Y$=INPUT$(1):PRINT Y$
390 PRINT CL$:IF Y$="Y"OR Y$="y"THEN 410 ELSE IF Y$="N"OR Y$="n"THEN 540
400 GOTO 380
410 PRINT"@1 (NAME)      "N$:PRINT"@2 (TITLE)     "T$:PRINT"@3 (COMPANY)   "C$
420 PRINT"@4 (ADDRESS)   "A$:PRINT"@5 (LOCATION)  "L$:PRINT
430 PRINT"PRESS @ TO BE CORRECTED OR <CR> IF OK ";:NC$=INPUT$(1):PRINT NC$:PRINT
440 IF NC$="1"THEN PRINT N$ ELSE IF NC$="2"THEN PRINT T$
450 IF NC$="3"THEN PRINT C$ ELSE IF NC$="4"THEN PRINT A$
460 IF NC$="5"THEN PRINT L$ ELSE IF NC$=CHR$(13) THEN 540
470 PRINT"IS THIS THE ENTRY TO BE CORRECTED? (Y/N):- ";:Y$=INPUT$(1):PRINT Y$
480 IF Y$="Y"OR Y$="y"THEN 500 ELSE IF Y$="N"OR Y$="n"THEN PRINT CL$:GOTO 410
490 GOTO 470
500 PRINT:PRINT"REENTER ENTIRE LINE INCLUDING CORRECTION:-";:LINE INPUT EC$
510 IF NC$="1"THEN N$=EC$ ELSE IF NC$="2"THEN T$=EC$ ELSE IF NC$="3"THEN C$=EC$
520 IF NC$="4"THEN A$=EC$ ELSE IF NC$="5"THEN L$=EC$
530 PRINT CL$:PRINT N$:PRINT T$:PRINT C$:PRINT A$:PRINT L$:GOTO 380
540 PRINT"DO YOU WANT TO CONTINUE? (Y/N):- ";:Y$=INPUT$(1):PRINT Y$:PRINT CL$
550 IF Y$="Y"OR Y$="y"THEN 90 ELSE IF Y$="N"OR Y$="n"THEN POKE 3,169:END
560 GOTO 540
```

As a supplement to SPW.BAS (see REMark Issue 35), here is a CP/M program that will permit you to directly address an envelope using a friction/traction type printer such as the Epson MX-80 FT. As an added feature, the label portion of this program has also been included.

This program is presented as strictly 'bare bones', there are no program comments. In contrast to SPW.BAS, this program grew to more than twice the length of the Single Page Writer. However, it does provide some very interesting features! In addition to the addressing of envelopes, provisions were made for the addressing of labels. One particularly significant aspect of this program is that its purpose is primarily for the home computerist. Since you use this program every time you want to print an address on an envelope or label(s), you DON'T need to accumulate a large data file. You use it like a typewriter except with word processing features. Also, since it is an MBASIC program, many of the parameters can be changed as desired.

Here are some of its features:

1. You can print any number of address lines from two up to five lines.

2. Upper or lower case characters or any combination can be used in any or all address lines.

3. For distinctive appearance, the first line of the address is emphasized.

4. All of the address lines are close spaced except the two line address. This was specifically designed to provide one space between the first and second line so it wouldn't look so skimpy.

5. Review and correction features have been incorporated to permit changes without the need to redo the entire address.

6. Any number of labels from one and on can be selected and reviewed or printed. Note: I prefer to print just one label at first for alignment purposes.

7. Once an address has been entered, you can review or print it for an envelope or

label(s), or make any changes to it. You can do any of these repeatedly until you decide to discontinue the program.

8. All address lines start at the same column position - all are in line. Hence the file 'ADDRINL.BAS' and the title 'ADDRESSING ENVELOPES/-INLINE'.

9. Label addresses are automatically centered on conventional 3 1/2 inch wide labels according to the longest line of the address.

10. Responses to entry or other function requests can be made with either upper or lower case characters.

In addition to the above, a surprise will be presented near the end of this article!

Now, since there are no comments within the program, a line by line description follows.

Line 10. Assigns the string variable CL$ to CLEAR SCREEN. Also, identifies the program file as 'ADDRINL.BAS'.

Line 20. Clears screen and displays program title.

Lines 30 to 80. Entries for all address information.

Line 90. Choice for either review or printout.

Line 100. Insures that the proper character for a review or printout had been selected, i.e., 'R', 'r', 'P', or 'p'.

Line 110. Assigns string variables EM$ and DM$ to emphasize and de-emphasize printing. These are for the Epson printer function codes.

Line 120. Nullifies string variables EM$ and DM$ to prevent catastrophic video graphic interference when in the review mode. The POKE 3,105 sets all LPRINT statements to go to the video display.

Line 130. Choice for either envelope or label(s).

Lines 140 to 150. Insures that the proper character had been selected for envelope or label(s), i.e., 'E', 'e', 'L', or 'l'.

Lines 160 to 180. Self explanatory. Also, the POKE 3,169 in line 180 lets the LPRINT statements to go to the printer.

Lines 190 to 220. Determines the numerical length of each line in the address and selects the longest for label centering.

Line 230. Sets the left hand margin for the address on the envelope (M=35) or makes necessary calculations to center the address on labels.

Note: You can change the address location on the envelope by changing the value for 'M'. M=35 is for a 9 1/2 inch envelope.

Line 240. If the label mode was selected, it starts the FOR/NEXT loop (S=0) for proper spacing between 15/16 inch labels and sets the counter (I=0 TO NL) for the number of labels requested.

Lines 250 to 320. Displays for review or prints out all the entered address lines.

Line 330. The IF/THEN must be true if you want to CORRECT any of the address lines. If it is not true, then it will ask if you want to continue. You MUST have selected both REVIEW and ENVELOPE modes to be able to make address line corrections.

Lines 340 to 370. Determines the proper spacing between 15/16 inch labels for any number of address lines.

Lines 380 to 400. Self explanatory.

Lines 410 to 530. Identifies and displays all entries so that line corrections can be made, if necessary. Verifies the line to be corrected and upon correction, displays the result.

Lines 540 to 560. Offers choice of continuing with the same address or terminating the program.

General comments: If you are interested in programming and whether you can use this program or not, I would like to suggest that you first READ it through and then follow the LOGIC of each mode of operation. There are any number of goodies imbedded in the program that could prove very useful in some of your future programs. Some of the goodies are brand new to me as I was obliged to develop them to accomplish a new function. I would also like to suggest that you experiment with this program to try out other ideas of your own and especially if you own a printer other than the MX-80 FT. Developing a new program for a useful purpose can be most educational and a lot of fun.

With ADDRINL.BAS loaded in MBASIC, perform the following:

```
1. Changes to lines:
   10  ADDRINL to ADDRCNT
   20  INLINE to CENTERED
  100  110 to 105
  170  190 to 230
  230  M=35 to M=45  and  M=INT(16-A/2) to M=16
  250  TAB(M) to TAB(M-C)
  280  TAB(M) to TAB(M-C)
  290  TAB(M) to TAB(M-N) and TAB(M) to TAB(M-T)
  300  TAB(M) to TAB(M-C)
  310  TAB(M) to TAB(M-A)
  320  TAB(M) to TAB(M-L)

2. Delete lines:
   190, 200, 210 and 220

3. Enter lines:
0) 105 M=INT(LEN(M$)/2):
            T=INT(LEN(T$)/2):C=INT(LEN(C$)/2):
   A=INT(LEN(A$)/2)
*) 110 L=INT(LEN(L$)/2):
            EM$=CHR$(27)+"E":DM$=CHR$(27)+"F":

4. Type RENUM <CR>

5. Type SAVE "ADDRCNT",A <CR>

0) Enter the two lines all on ONE continuous line.

*) It is necessary only to insert
   'L=INT(LEN(L$)/2):'
   in existing line 110.
```

You may not favor ADDRCNT.BAS just as I didn't, at first. I was particularly unimpressed with the labels for mailing purposes. I never really liked using labels for personal correspondence. A label somehow makes it look cold and impersonal. This is probably due to my having received too many pieces of junk mail with labels on them. On the other hand, I don't mind using labels on packages. However, I did discover some very useful purposes for the centered type labels along with a fringe benefit - making my wife happy!

If you don't mind fooling your computer when it asks you to 'ENTER Person's Name', etc., you can surreptitiously enter such items as 'DILL PICKLES', 'PEACH PRESERVES', etc. Of course, you could make these into multiple line labels to include date put up and by whom. Other uses could be for loose leaf covers, identification for many kinds of storage boxes and a lot more. The nice part of this program is that you can easily print as many or as few special labels as you want whenever you want them. I particularly like the centered type labels for these purposes.

In conclusion, I'd like to encourage everyone interested in computering to share your ideas with the rest of us by writing up your pet programs or interesting goodies and send them in to HUG for publication. I am sure you will find the effort most gratifying in many ways. Happy Computering!

HUG NEW PRODUCTS

885-1126
HDOS UTILITIES by PS: $20.00

Introduction: This disk contains a collection of HDOS utility programs for listing files on a disk, examining text files, and testing memory.

Requirements: These programs require the HDOS operating system version 1.5 to 2.0 and will run on any H8/H17 or H/Z-89,90 with at least 32k of memory. The H/Z-19 or H/Z-29 terminal is required to use the DIR19 and SEE programs with an H8. The SEE program will also work on H/Z-100 computers under HRUN. The ALLRAM program is only for ORG-0 compatable H/Z-89,90 computers with 64k of RAM.

The following files are included on the disk.

README	.DOC	SEE	.ABS
DIR19	.ABS	SEE	.ASM
DIR19	.ASM	PPRT	.ABS
PDIR	.ABS	PPRT	.ASM
PDIR25	.ABS	ALLRAM	.ABS
PDIR	.ASM	ALLRAM	.ASM

Author: All programs are by Patrick Swayne, HUG.

DIR19 - This is a program for displaying the files on a disk that takes advantage of Heath/Zenith terminal features to list as many files as possible on the screen in an easy-to-read format. In addition to file names, the size in sectors and the flags for each file are shown. DIR19 displays up to 80 files on the screen, and if there are more than 80 to show, it prompts the user to hit RETURN to show another "page" of files, and will continue this process for up to 255 files. Files can be listed alphabetically or in their actual directory order. DIR19 can take "wild card" arguments to allow the user to show specific groups of files or individual files.

DIR19 divides the screen into 6 sections using graphic lines. At the top of the screen, a one-line section holds the drive name (SY0:, etc.), the volume number, and the disk label. In the middle of the screen are four vertical blocks that hold up to 20 files each. DIR19 fills each block from top to bottom in turn until all files requested are displayed or the screen fills up. At the bottom of the screen, a one-line section displays the count of files shown, the total size of the files, and the free space on the disk. The user can include switches in the command line to show system files, show allocated disk usage, and/or to suppress alphabetizing.

PDIR and PDIR25 - This program works like DIR19, but its output goes to a printer. PDIR25 uses H/Z-25 graphics, while PDIR is for any printer.

SEE - This program is a replacement for the TYPE command normally used to examine text files that uses Heath/Zenith terminal features. It was inspired by a similar program called "SC" by John Stetson, but was developed independently. With SEE, the user can press function keys to scroll forward or backward in a text file by lines or 24-line pages, jump immediately to the top or bottom of a file, search for words or phrases in the file, or print individual screens on a printer. It also provides horizontal scrolling so that lines longer than 80 characters can be viewed. Files that are too large to fit in available memory can be viewed in segments. SEE counts the lines in a file and displays the number of the line that is currently at the top of the screen on the 25th line, along with key prompts and the name of the file. The line numbers are maintained sequentially when you view large files that take up more than one memory segment.

SEE automatically determines what kind of computer or terminal it is being used on and optimizes its operation for that particular computer or terminal. It also makes sure that the key prompts displayed on the 25th line match the computer or terminal's function keys. On H/Z-29 terminals and H/Z-100 computers, the prompts are labeled since the relationship between the keyboard and screen may not be fixed.

Note: SEE was written for use with both HDOS and CP/M. It will eventually be released on a HUG CP/M disk, but you can use the HDOS version on CP/M by copying the file SEE.ASM to a CP/M disk with a utility such as HTOC (885-1207) and re-assembling it with the CP/M assembler.

PPRT - The Push-Pop RAM Test is a special memory test designed to help you find speed sensitive memory failures. It uses the processor's PUSH and POP instructions for reading and writing memory, which are more taxing on slow components than other instructions. This test may be useful if you performed one of the speed modifications detailed in REMark Issues 34 and 38.

Note: Like SEE, PPRT was written for use with both HDOS and CP/M. You can copy PPRT.ASM to a CP/M disk and re-assemble it for use with CP/M.

ALLRAM - This program allows an H/Z-89 or Z90 user with 64k of RAM and ORG-0 capability to access all of his computer's memory as RAM under HDOS. It frees up a small area that can be used for such things as USR space for MBASIC, etc. It also helps in trouble shooting speed modification problems.

885-1230 [-37]
KEYMAP
Function Key Mapper $20.00

Introduction: KEYMAP is a program that lets the user define the

computer's function keys and keypad keys. Once defined, the keys will send the sequence of characters the same as if the characters were typed from the keyboard. Versions are included for using KEYMAP with BASIC and Wordstar.

Requirements: KEYMAP requires the CP/M operating system and runs on any Heath or Zenith desk top computer that can run standard CP/M, including the H/Z-89, 90 and H/Z-100. An H/Z-19 or H/Z-29 terminal is required for use with an H8/H17 computer.

Instructions are included for implementing KEYMAP with your computer system.

The following files are included on the HUG P/N 885-1230 [-37] KEYMAP disk.

README	.DOC	KEYWS	.COM
KEYMAP	.COM	KEYWS	.DOC
KEYMAP	.ASM	KEYWS	.89
KEYMAP	.DOC	KEYWS	.100
STATLIN	.COM	KEYWS	.29
STATLIN	.ASM	KEYSYS	.COM
KEYBAS	.COM	KEYSYS	.DOC
KEYBAS	.DOC	KEYSYS	.89
KEYBAS	.89	KEYSYS	.100
KEYBAS	.100	KEYSYS	.29
KEYBAS	.29		

Author: All programs are by P. Swayne, HUG.

Program Content: KEYMAP is a program that lets you define the characters that are produced by your computer's function and keypad keys. Up to ten characters can be produced by each mappable key, including control characters. The keys that can be mapped are the ERASE (or F0) key, the F1 through F8 keys (H19 color keys), the keypad 1 through 8 keys, and also the separate arrow and HOME keys if you have an H/Z-100 or H/Z-29. You can define any one of these keys as a "function shift" key, which allows every other mapped key to produce two responses each. If you wish, KEYMAP can place a message on the status line (25th line) of your screen indicating the responses of the function keys. KEYMAP becomes part of "the system" when it is loaded so that you can run any program with it to take advantage of the mapped keys. Provision is made to temporarily disable mapping and even to run more than one KEYMAP at a time to provide special responses for different programs.

KEYMAP has a special set up mode that makes it easy to define your own responses for each mapped key. Pre-configured versions are included for use with BASIC and WordStar, along with a version for general CP/M use.

KEYMAP.COM - The KEYMAP program in unconfigured form (key responses are not altered).

KEYMAP.ASM - The source code for KEYMAP.COM

KEYMAP.DOC - Instructions for configuring and using KEYMAP.

STATLIN.COM - A program to generate status line (25th line) messages that can be used by KEYMAP.

STATLIN.ASM - The source for STATLIN.COM.

KEYBAS.COM - A pre-configured version of KEYMAP that produces BASIC keywords (PRINT, GOTO, etc.) when function and keypad keys are pressed. 37 different keywords are instantly available to simplify entry of BASIC programs.

KEYBAS.DOC - Instructions for using KEYBAS.

KEYBAS.89, KEYBAS.100, KEYBAS.29 - Status line messages for use with KEYBAS.COM. KEYBAS.DOC explains which one to use for your system.

KEYWS.COM - A pre-configured version of KEYMAP that produces WordStar control characters to allow cursor movement with the arrow keys and easy use of many WordStar features. Indenting, centering, paragraph reform, and file scrolling are only a few of the functions mapped to function and keypad keys.

KEYWS.DOC - Instructions for using KEYWS.

KEYWS.89, KEYWS.100, KEYWS.29 - Status line messages for use with KEYWS.COM.

KEYSYS.COM - A pre-configured version of KEYMAP that produces CP/M system commands (DIR A:, PIP, etc.) when function keys are pressed. The RETURN code is included in each response so that the functions are acted upon immediately when you press the appropriate key.

KEYSYS.DOC - Instructions for using KEYSYS.

KEYSYS.89, KEYSYS.100, KEYSYS.29 - Status line messages for use with KEYSYS.COM.

Comments: The pre-configured KEYMAP programs are only a sample of what you can do with KEYMAP. It brings the power of programmable function keys to every Heath/Zenith CP/M user.

885-8015
HDOS TEXTSET Formatter $30.00

Introduction: TEXTSET is an interactive formatter that will take a text file and format it to the users specifications. It has a number of features which are explained below. TEXTSET is used with a Diablo Printer to produce high quality printing for any size file. The special features of the Diablo printer are controlled by the program. No hardware modification needs to be done.

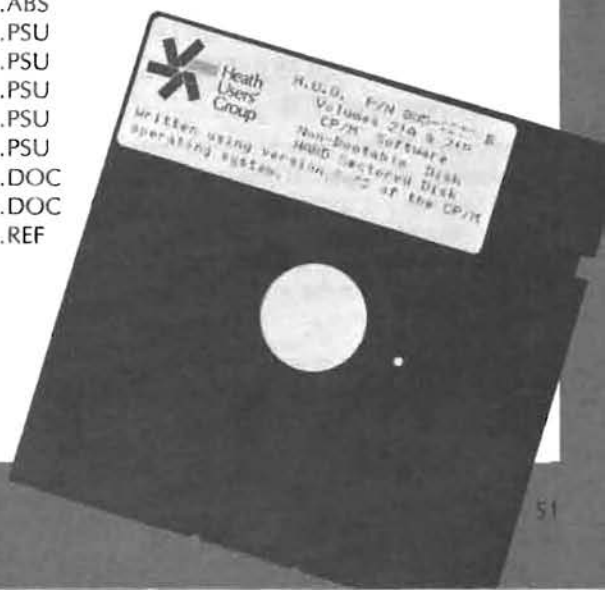
Requirements: TEXTSET requires the HDOS operating system version 2.0 (but should run on 1.6) on an H8/H17/H19 or H/Z89 with 56K of memory. Only one disk drive is required, however, two drives are recommended.

Note: A Diablo printer 1640 ksr, 1630, or RO 630 (Heath # WH-54) is required for use with TEXTSET.

The user will also need a text editor (e.g. HUG P/N 885-1022) or a word processing system that runs under HDOS. If you are not familiar with or do not have a favorite editor, the HDOS EDIT program comes with the HDOS operating system and has instructions in the HDOS manual.

The following files are contained on the HUG P/N 885-8015 HDOS TEXTSET Formatter disk:

TEXTSET	.ABS
TEXTSET	.PSU
PICA	.PSU
ELITE	.PSU
PLASTIC	.PSU
METAL	.PSU
DEMO1	.DOC
DEMO2	.DOC
TEXTSET	.REF



Author: Terry W. Wilk

Program Content: TEXTSET will format a text file created from an editor and print it to a Diablo printer. It will do microspace justification, proportional spacing of the characters, automatic formatting of the text within a selectable width, double striking, bold striking, underlining, centering, and right justification of lines. No special print wheels are necessary. However, the selectable proportional spacing can be used effectively with special print wheels.

TEXTSET requires interaction with the user to produce a final text format. It is designed this way to give the user flexibility in working with formatting text. The program pauses for each question and allows the user to make any of a number of selectable options. TEXTSET allows the user to select:

1) the PSU (Proportional Spacing Unit) Table
2) the WIDTH of formatted text
3) the OFFSET (amount of space between characters)
4) the SPACE (PSU for the space character)
5) the BLANK MAX (the maximum space between words, before adding tiny spaces between characters)
6) the LINES (total lines per block to be processed)
7) the FORCE BR (set a force break after each line)
8) SET VMI (change the line feed value of the Diablo)
9) SET LEFT MARGIN
10) LINE FEED (send any number of line feeds to the printer)
11) SEND LINE (type and send a line, e.g. a header or page no.)
12) EXIT, RESTART, PROCESS, OUTPUT, SKIP, and CHANGE

TEXTSET will not automatically do page numbering or auto headers/footers. These features are possible, however, through the SEND LINE command of TEXTSET.

There are other features of TEXTSET which make this package useful. TEXTSET checks for user input errors and errors in the input text file. Predefined PSU (Proportional Spacing Unit) files are included on the disk. The user can customize or create PSU Tables to any spacing unit.

Most importantly, the documentation is very user friendly and contains step by step instructions on the use of TEXTSET. Two example files and one reference file (DEMO1.DOC, DEMO2.DOC, and TEXTSET.REF, respectively) are explained and processed through the instructions. The documentation also explains how to implement TEXTSET with one or two disk drives.

Comments: TEXTSET is a must for any user who needs a formatter that will use the special features of the Diablo printer to produce a professional formatted text file. The documentation is one of the best instructions manual that the reviewer has seen.

885-8016
HDOS MORSE CODE
Transceiver Ver 2.0 . . . . . $20.00

Introduction: This is an all new major upgrade of the previously released Version 1.1 under part number 885-1016. Many new user friendly improvements and extensive features have been added. The program allows the user to change custom data (e.g. the station call sign) at any time from the keyboard. Some of the other new features are listed below.

MORSE CODE Transceiver version 2.0 is an 8080 assembly language program which provides the operator with the ability to send or receive morse code. The program is intended to be used by Amateur Radio Operators to facilitate communication by morse code over a wide range of code speeds, dot/dash ratios, interference, and noise conditions. In addition, the precision speed feature is intended to be used whenever extreme transmit code speed accuracy is required.

The precision morse code speed algorithms used in this program were originally developed as part of a set of custom H89 programs for the American Radio Relay League's Maxim Memorial Station 'W1AW' at A.R.R.L. Headquarters in Newington, CT.

Requirements: This program requires the HDOS operating system version 2.0 on an H19/H8/H17 or H89 with 48K of memory. Only one drive is required, however, two are recommended.

All I/O is at RS232C levels via the DTE port. External equipment is required to interface the RS232C level I/O signals to the amateur station equipment. Design details were published in REMark Issue 33, October 1982, page 17.

Note: The algorithms used for morse code decoding depend on timing from the internal clock. Therefore, this program will perform properly only on a standard machine running at 2.048 MHz.

The following files are included on the HUG P/N 885-8016 MORSE CODE Transceiver disk:

README	.DOC	TXLOP	.ACM
CW	.DOC	TXSPEED	.ACM
CW	.ASM	RXINT	.ACM
CW	.ABS	RXLOP	.ACM
CWDATA	.DAT	DISPLAY	.ACM
SAVMSG	.DAT	STATDPY	.ACM
KBIN	.ACM	CODETBL	.ACM
SPKEY	.ACM	VARTBL	.ACM

Some additional HDOS 2.0 XTEXT files are required for assembling CW.ASM.

Author: Robert R. Anderson K2BJG

Program Content: MORSE CODE Transceiver can receive and transmit standard morse characters as well as special morse characters such as (AR), (SK), (BK), (KN), (BT), AND (AS). Both upper and lower case key input and screen display is allowable.

The CWDATA.DAT disk file contains eleven multi-character groups which can at any time be read to the currently selected buffer and display screen. This file contains commonly used abbreviations and space for the station call sign and station location.

Two transmit buffers selected by the 'f2' key will transmit up to 254 characters. Ten message buffers can be loaded or cleared under control of the 'f1' key. Any selected message buffer can be transferred to the transmit buffer. These 10 buffers are saved in the SAVMSG.DAT.

Disks can be changed and ASCII disk files can be loaded without leaving the program. The memory buffer extends from the end of the program to the top of available memory.

The receive program operates in three modes: LOCK, TRACK, and HOLD. The transmit program operates in two modes: NORMAL and DISK FILE. The modes can be manually switched to any of the modes.

The available precision fixed Tx speeds are: 5SP, 05, 7.5, 10, 13, 15, 18 WPM, and 20 through 70 WPM in 5 WPM steps. The morse code speed standard used in fixed speed mode is in accordance with amateur practice of one word being defined as consisting of 50 elements.

The screen display is split into three areas for viewing the receive

or transmit data, the selected transmit pre-type data, and the selected message buffer data. The screen is also used as the command screen and for display of error messages. The program will not allow for improper keyboard commands to take place.

The documentation is contained on the disk. A listing of the instructions will be recommended for learning the MORSE CODE Transceiver program.

Comments: No comments.

885-8017
HDOS Programmers Helper $16.00

Introduction: The H89 Programmers Helper (PH.DVD) is an interrupt device driver that may be called from the command or program mode of Benton Harbor BASIC, EDIT, DBUG, or most any other host program by typing the BREAK key. PH.DVD will perform any of its functions and then return control to the host program. The functions are listed below.

Requirements: This program requires the HDOS operating system version 2.0 with an H19/H8/H17 or H89 with a minimum amount of memory. Only one drive is required.

Note: The H/Z-19 terminal is required to run PH.

The following programs are included on the HUG P/N 885-8017 HDOS Programmers Helper disk:

README	.DOC
PH	.DVD
PH	.ASM

Author: Richard H. Livingston

Program Content: PH.DVD will perform a number of functions to aid the programmer. It is invoked by entering the BREAK key or a CTRL-@. The functions that it will perform are as follows:

1) Number Base conversions: convert between Binary, Octal, Split-Octal, Decimal, Hexidecimal, and ASCII

2) ASCII conversion: echo the value for the key depressed

3) Flag conversion: convert the numerical value of the Flag register to the logical state of each Flag

4) DCA conversion: line and column numbers are converted to the proper ASCII escape sequence for direct cursor addressing

5) 16 bit positive integer arithmetic: addition, subtraction, multiplication, and division

6) Logic functions: 1's Cmp, 2's Cmp, AND, OR, XOR

7) Graphic mode: displays the graphic character set and associated key characters on the 24 and 23 lines, respectively

A second BREAK key or CTRL-@ will return control the host program. PH.DVD must be loaded before it can be used. The instructions are included on the disk.

Comments: This device driver provides the programmer with an excellent tool for doing number base conversion and other functions at the press of the BREAK key.

The following five HDOS products are available in soft-sectored format beginning this month:

885-1078 [-37] HDOS Z80 Assembler
885-1107 [-37] HDOS Data Base System H8/H89
885-1038 [-37] Wise on Disk H8/H89
885-1042 [-37] PILOT on Disk H8/H89
885-1064 [-37] Disk IX H8/H89 Disk

HUG Price List

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

Part Number	Description of Product	Selling Price	REMark Issue
HDOS			
885-1029 [-37]	Disk II Games 1 H8/H89	$ 18.00	40
885-1060 [-37]	Disk VII H8/H89	$ 18.00	40
885-1062 [-37]	Disk VIII H8/H89 (2 Disks)	$ 25.00	40
885-1067 [-37]	Disk XI H8/H89 Games	$ 18.00	40
885-1071 [-37]	MBASIC SmBusPk H8/H19/H89	$ 75.00	41
885-1086 [-37]	Tiny HDOS Pascal H8/H89	$ 20.00	40
885-1089 [-37]	Disk XVIII Misc H8/H89	$ 20.00	41
885-1090 [-37]	Disk XIX Utilities H8/H89	$ 20.00	41
885-1097 [-37]	MBASIC Quiz Disk H8/H89	$ 20.00	41
885-1108 [-37]	HDOS MBASIC Data Base System	$ 30.00	41
885-1121	Hard Sectored Support Package	$ 30.00	37
885-1122	MicroNET Connection	$ 16.00	37
885-1123	XMET Robot/Cross Assembler	$ 20.00	40
885-1124	HUGMAN & Movie Animation Pkg	$ 20.00	41
885-1125	MAZEMADNESS	$ 20.00	41
885-8016	Morse Code Transceiver Ver 2.0	$ 20.00	41
CP/M			
885-1211 [-37]	Sea Battle	$ 20.00	36
885-1222 [-37]	Adventure	$ 10.00	36
885-1223 [-37]	HRUN HDOS Emulator	$ 40.00	37
885-1224 [-37]	MicroNET Connection	$ 16.00	37
885-1225 [-37]	Disk Dump and Edit Utility (DDEU)	$ 30.00	38
885-1226 [-37]	CP/M Utilities by PS:	$ 20.00	38
885-1227 [-37]	CP/M Cassino Graphic Games	$ 20.00	38
885-1228 [-37]	CP/M Fast Action Games	$ 20.00	39
885-1229	XMET Robot/Cross Assembler	$ 20.00	40
885-1229 [-37]	XMET Robot/Cross Assembler	$ 20.00	40
885-3003 [-37]	ZTERM Modem Package	$ 20.00	36
885-8012 [-37]	Modem Appl. Effector (MAPLE)	$ 35.00	36
ZDOS			
885-3004-37	ZBASIC Graphic Games Disk	$ 20.00	37
885-3005-37	ZDOS ETCHDUMP	$ 20.00	39
MISCELLANEOUS			
885-0004	HUG 3-Ring Binder	$ 5.75	
885-4001	REMark VOLUME 1, Issues 1-13	$ 20.00	
885-4002	REMark VOLUME 2, Issues 14-23	$ 20.00	
885-4003	REMark VOLUME 3, Issues 24-35	$ 20.00	
885-4600	Watzman/HUG ROM	$ 45.00	41

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

Jumbo Letters and Numbers

Programming Graphic Shapes

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

This month's column will provide two sub-programs for your library. We'll make use of them in some short routines, and will show you how to add them to your own programs. The numbering shown for the program lines may seem a bit odd, but type them as they're shown anyway. The reason we've done this is to let us merge them into a larger program while having a standard line number to which we can direct processing for each function. It will be the beginning of your modular programming.

We're going to build four-character-high letters and numbers, and we'll generate each character by way of a separate sub-routine. This allows any character to be called upon or changed without affecting the others. When this is done, we'll add control information to permit typing of these characters from a program.

What does it take to generate a letter four characters high? Well, first we've got to know what shape we want, then we've got to use the graphic symbols available to us to build that shape. Those of you who have read my past articles probably already have your computer manual at your side. For those of you who are new to my style of writing, I believe in knowing those manuals inside out. I won't write something without telling you where to find the necessary information, but I expect you to look it up. Past experience shows that the difference between fighting with and enjoying your computer most often lies in an understanding of the documentation your computer's manufacturer spent all that time and money to produce. Besides, this column is supposed to provide a learning experience, and there's nothing more important to learn than good habits. Knowing your manual is one such habit, so take a break, relax, and dig that manual out of the closet.

Now that we have our manuals, let's look at the graphic symbols available to us. In my manual, it's chapter 12.

To digress a moment now that you've got books open everywhere, here's a hint I found helpful. First, I cut the binding off my manual. I used a power saw to shave off 1/8 inch of the back edge. Then I punched the pages for a three-ring binder. I really think this is how Heath should furnish their operating manuals, but I guess it's a cost thing. Anyway, it lays flat and handles much easier this way.

Back to our big letters. As we look at the graphic symbols available to us, we see a character and two numbers next to each of them. We must send one of these pieces of information to the terminal (the section that controls the screen) while in the Graphics Mode to generate that particular symbol. I explained all that in my prior article "Screen Control", so you may want to refer to that to fully understand what's going on. If you don't have it, you can get by without it, but an order to HUG for the back issue is the best advice I can give. Every issue of REMark I've ever read gave me at least some valuable information, and I've got them all.

Let's lay out the letter A. Look at Figure 1. It's a grid four characters high and four characters wide. In it I've sketched the individual graphic symbols we'll need to make our big A. We can see that we'll have to print four separate lines, but first we must find each symbol in our manuals. And wouldn't you know it, the first one doesn't exist. There's no graphic symbol for it. However, a lower case "r" is just the opposite of what we want. If we could print the blank part, and leave blank the printed part, that would be perfect. Well, that's just what REVERSE VIDEO does. To make that first symbol on the first line of our big A, we must:

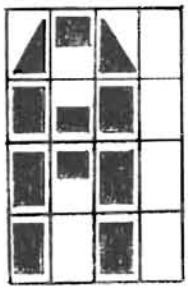


Figure 1

```
ENTER REVERSE VIDEO
PRINT A  LOWER CASE  "r"
```

This is where our control program "CONTROL.BAS" from last month's column comes in. For now, let's assume that the terminal is already in its GRAPHICS MODE. Using the sub-program CONTROL.BAS, we enter REVERSE VIDEO with the program line:

```
PRINT E9$
```

We then print our symbol with:

```
PRINT "r"
```

With that done, we look for the next symbol. It's produced by the letter "p", but this time not in REVERSE VIDEO. To exit this mode, we again use last months program and add the line:

```
PRINT E10$
```

Then make the symbol with:

```
PRINT "p"
```

The final symbol of the first line is the opposite of the symbol generated by an underscore, which is also ASCII Character Number 95. So, again enter REVERSE VIDEO with:

```
PRINT E9$
```

Then print the symbol.

```
PRINT CHR$(95)
```

Finally, get back out of REVERSE VIDEO with:

```
PRINT E10$
```

Now, let's put that all together on one program line.

```
5000 PRINT E9$;"r";E10$;"p";E9$;CHR$(95);E10$
```

You've just built the top line of a big A. You've made a character with graphic symbols. While it seemed slow and cumbersome, that was just because we took time to explain each step. It goes much faster after you've done it a few times. Also, there are excellent graphic layout sheets produced by several vendors. These make it fast and easy to lay out the shapes you want, and to pick the character which makes those shapes.

The last three lines needed to complete our letter A are as follows. I leave it up to you to look up the reason we use the symbols we do. Each of these lines works like the first one. Only the shapes and the characters that generate them are different. Here are the lines.

```
5001 PRINT "i";E9$;"p";E10$;"i"
5002 PRINT "ipi"
5003 PRINT "i i"
```

Now we'll experiment with these four lines and improve them. Load CONTROL.BAS from the last month's column and add the four lines above. Then add these three lines.

```
500 PRINT E7$    'ENTER GRAPHICS MODE
6000 PRINT E8$   'EXIT GRAPHICS MODE
6010 END
```

With that done, type RUN and watch a big "A" appear.

So far, so good. We have written a program which produces a large shape in graphics. It works fine as long as we want that shape on the left of the screen. To put the character anywhere we want it, we'll have to use Direct Cursor Addressing. As you recall from the "Screen Control" article, that sends the cursor anywhere on the screen. By sending the terminal the right instruction with information about the line number and the space number we want, we can print anywhere. Each number must be increased by 31. Refer to your manual, Chapter 12, Direct Cursor Addressing, and to my previous article if you have any doubt about how this works.

We've provided for the Direct Cursor Addressing function in CONTROL.BAS, so let's expand our four program lines to use it. We'll use the variables Y to be our line number plus 31, and X to be the space number plus 31. Our letter A now becomes:

```
5000 PRINT E1$;CHR$(Y);CHR$(X);E9$;
                                "r";E10$;"p";E9$;CHR$(95);E10$
5001 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"p";E10$;"i"
5002 PRINT E1$;CHR$(Y+2);CHR$(X);"ipi"
5003 PRINT E1$;CHR$(Y+3);CHR$(X);"i i"
```

All we have to do is specify the upper left-hand corner of the letter and it will fall into place. Each row of graphic symbols prints one line lower because we have added to the value of Y. The value of X stays the same and aligns each row directly below the previous one. To see how this works, fire up your computer and RUN MBASIC. Then LOAD "CONTROL.BAS", add the four lines above, and the following six lines.

```
500 PRINT E7$          'ENTER GRAPHICS MODE
510 PRINT E2$          'CLEAR THE DISPLAY
520 Y=41               'VERTICAL POSITION TO LINE 10
530 X=68               'HORIZONTAL POSITION TO COLUMN 37
6000 PRINT E8$         'EXIT GRAPHICS MODE
6010 END
```

Type RUN and watch what happens. Now try different values for X and Y. You can do that by retyping just lines 520 and 530. To pick the value you want, just add 31 to the line number for Y and to the column number for X. You've come a long way and have earned a chance to play and take a break. Who said learning can't be fun?

So far, we've seen how to build a character using Heath graphic symbols, and how to print that character anywhere on the screen. Now we'll add two lines of programming to make our work really useful. The first line will move the cursor to the right of our letter so it's in the correct place for the next letter to begin, and will provide a control character "C" which tells our program how wide the letter is. We'll find this useful for things like backspacing later on. This feature will be used in the game called WORDS that we'll develop next month. The line we add is:

```
5004 X=X+4:C=4
```

Finally, one word on line 5005 makes this a truly independent subroutine.

```
5005 RETURN
```

Now run this final test.

LOAD "CONTROL.BAS"

Add lines 5000 thru 5005 above.

Add the following lines.

```
500 PRINT E2$     'CLEAR THE DISPLAY
510 PRINT E7$     'ENTER GRAPHICS MODE
520 Y=39          'LINE 8
530 X=61          'COLUMN 30
540 GOSUB 5000    'GO PRINT AN "A"
550 GOSUB 5000    'GO PRINT ANOTHER "A" CORRECTLY SPACED
560 GOSUB 5000    'PRINT A THIRD "A"
570 Y=47:X=61     'LINE 16, COLUMN 30
580 GOSUB 5000:GOSUB 5000:GOSUB 5000 'PRINT THREE "A"'S
590 PRINT E8$     'EXIT GRAPHICS MODE
600 END
```

You should have three rows of A's. One row was produced by separate lines of programming, and the other by combining instructions on a single line. This brings us to the end of this month's column. Let's review what we've learned.

1) How to build a shape or character from Heath Graphic Symbols.
2) How to control the location of this character on the screen.
3) How to build an independent sub-routine which may be called by your program, be executed, and return to the program.
4) One way to call a sub-program we saved before, and use it.

Before I quit, I'm going to leave you with two new programs for your library. We'll use one of them next month as we begin our game program. We'll also be sneaking in some CAI (Computer Aided Instruction) in a game that's fun for the whole family. Type these programs at your leisure. Because they can be used by any BASIC program by way of GOSUB commands, you should save them in the ASCII format. This is necessary so we can add them to our other programs with the MERGE command. This is the only time you'll ever have to type them. To save in ASCII, use the command line:

SAVE "LETTERS",A

Here are the two programs. See you next month.

```
10 ********** LETTERS.BAS **********
20 ********** COPYRIGHT 1982 *********
30 ******** APPLIED COMPUTING ********
5000 PRINT E1$;CHR$(Y);CHR$(X);E9$;
                    "r";E10$;"p";E9$;CHR$(95);E10$  'A
5001 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"p";E10$;"i"
5002 PRINT E1$;CHR$(Y+2);CHR$(X);"ipi"
5003 PRINT E1$;CHR$(Y+3);CHR$(X);"i i"
5004 X=X+4:C=4
5005 RETURN
```

```
5010 PRINT E1$;CHR$(Y);CHR$(X);"ip";E9$;CHR$(95);E10$            'B
5011 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"p";E10$;"r"
5012 PRINT E1$;CHR$(Y+2);CHR$(X);"ip";E9$;CHR$(95);E10$
5013 PRINT E1$;CHR$(Y+3);CHR$(X);"i";E9$;"p";E10$;"r"
5014 X=X+4:C=4
5015 RETURN
5020 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$  'C
5021 PRINT E1$;CHR$(Y+1);CHR$(X);"i p"
5022 PRINT E1$;CHR$(Y+2);CHR$(X);"i ";E9$;"p";E10$
5023 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5024 X=X+4:C=4
5025 RETURN
5030 PRINT E1$;CHR$(Y);CHR$(X);"ip";e9$;CHR$(95);E10$            'D
5031 PRINT E1$;CHR$(Y+1);CHR$(X);"i i"
5032 PRINT E1$;CHR$(Y+2);CHR$(X);"i i"
5033 PRINT E1$;CHR$(Y+3);CHR$(X);"i";E9$;"p";E10$;"r"
5034 X=X+4:C=4
5035 RETURN
5040 PRINT E1$;CHR$(Y);CHR$(X);"ipp"                             'E
5041 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"pp";E10$
5042 PRINT E1$;CHR$(Y+2);CHR$(X);"ipp"
5043 PRINT E1$;CHR$(Y+3);CHR$(X);"i";E9$;"pp";E10$
5044 X=X+4:C=4
5045 RETURN
5050 PRINT E1$;CHR$(Y);CHR$(X);"ipp"                             'F
5051 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"pp";E10$
5052 PRINT E1$;CHR$(Y+2);CHR$(X);"ipp"
5053 PRINT E1$;CHR$(Y+3);CHR$(X);"i"
5054 X=X+4:C=4
5055 RETURN
5060 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$  'G
5061 PRINT E1$;CHR$(Y+1);CHR$(X);"i p"
5062 PRINT E1$;CHR$(Y+2);CHR$(X);"il";E9$;"p";E10$
5063 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";e10$;"r"
5064 X=X+4:C=4
5065 RETURN
5070 PRINT E1$;CHR$(Y);CHR$(X);"i i"                             'H
5071 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"p";E10$;"i"
5072 PRINT E1$;CHR$(Y+2);CHR$(X);"ipi"
5073 PRINT E1$;CHR$(Y+3);CHR$(X);"i i"
5074 X=X+4:C=4
5075 RETURN
5080 PRINT E1$;CHR$(Y);CHR$(X);"i"                               'I
5081 PRINT E1$;CHR$(Y+1);CHR$(X);"i"
5082 PRINT E1$;CHR$(Y+2);CHR$(X);"i"
5083 PRINT E1$;CHR$(Y+3);CHR$(X);"i"
5084 X=X+2:C=2
5085 RETURN
5090 PRINT E1$;CHR$(Y);CHR$(X);"  i"                             'J
5091 PRINT E1$;CHR$(Y+1);CHR$(X);"  i"
5092 PRINT E1$;CHR$(Y+2);CHR$(X);"  i"
5093 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5094 X=X+4:C=4
5095 RETURN
5100 PRINT E1$;CHR$(Y);CHR$(X);"i i"                             'K
5101 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"r";E10$;"r"
5102 PRINT E1$;CHR$(Y+2);CHR$(X);"i";CHR$(95);E9$;CHR$(95);E10$
5195 RETURN
5200 PRINT E1$;CHR$(Y);CHR$(X);"i i"                             'U
5201 PRINT E1$;CHR$(Y+1);CHR$(X);"i i"
5202 PRINT E1$;CHR$(Y+2);CHR$(X);"i i"
5203 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5204 X=X+4:C=4
5205 RETURN
5210 PRINT E1$;CHR$(Y);CHR$(X);"i  i"                            'V
5211 PRINT E1$;CHR$(Y+1);CHR$(X);"i  i"
5212 PRINT E1$;CHR$(Y+2);CHR$(X);CHR$(95);E9$;CHR$(95);"r";E10$;"r"
5213 PRINT E1$;CHR$(Y+3);CHR$(X);" ";CHR$(95);"r"
5214 X=X+5:C=5
5215 RETURN
5220 PRINT E1$;CHR$(Y);CHR$(X);"i   i"                           'W
5221 PRINT E1$;CHR$(Y+1);CHR$(X);"i   i"
5222 PRINT E1$;CHR$(Y+2);CHR$(X);"i";E9$;"r";E10$;"i";E9$;CHR$(95);
E10$;"i"
5223 PRINT E1$;CHR$(Y+3);CHR$(X);"ir ";CHR$(95);"i"
5224 X=X+6:C=6
5225 RETURN
5230 PRINT E1$;CHR$(Y);CHR$(X);"i i"                             'X
5231 PRINT E1$;CHR$(Y+1);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5232 PRINT E1$;CHR$(Y+2);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$
5233 PRINT E1$;CHR$(Y+3);CHR$(X);"i i"
5234 X=X+4:C=4
5235 RETURN
5240 PRINT E1$;CHR$(Y);CHR$(X);"i i"                             'Y
5241 PRINT E1$;CHR$(Y+1);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5242 PRINT E1$;CHR$(Y+2);CHR$(X);" i"
5243 PRINT E1$;CHR$(Y+3);CHR$(X);" i"
5244 X=X+4:C=4
5245 RETURN
5250 PRINT E1$;CHR$(Y);CHR$(X);"iii"                             'Z
5251 PRINT E1$;CHR$(Y+1);CHR$(X);" ";E9$;"r";E10$;"r"
5252 PRINT E1$;CHR$(Y+2);CHR$(X);E9$;"r";E10$;"r"
5253 PRINT E1$;CHR$(Y+3);CHR$(X);"iii"
5254 X=X+4:C=4
5255 RETURN


10 ********** NUMBERS.BAS **********
20 ********* COPYRIGHT 1982 *********
30 ******** APPLIED COMPUTING ********
6000 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"i"                  'i
6001 PRINT E1$;CHR$(Y+1);CHR$(X);" i"
6002 PRINT E1$;CHR$(Y+2);CHR$(X);" i"
6003 PRINT E1$;CHR$(Y+3);CHR$(X);"iii"
6004 X=X+4:C=4
6005 RETURN
6010 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$  '2
6011 PRINT E1$;CHR$(Y+1);CHR$(X);"  i"
6012 PRINT E1$;CHR$(Y+2);CHR$(X);" ";E9$;"r";E10$;"r"
6013 PRINT E1$;CHR$(Y+3);CHR$(X);E9$;"r";E10$;"ii"
6014 X=X+4:C=4
6015 RETURN
6020 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$  '3
6021 PRINT E1$;CHR$(Y+1);CHR$(X);" ";E9$;"p";E10$;"r"
```

```
5103 PRINT E1$;CHR$(Y+3);CHR$(X);"i i"
5104 X=X+4:C=4
5105 RETURN
5110 PRINT E1$;CHR$(Y);CHR$(X);"i"                                    'L
5111 PRINT E1$;CHR$(Y+1);CHR$(X);"i"
5112 PRINT E1$;CHR$(Y+2);CHR$(X);"i"
5113 PRINT E1$;CHR$(Y+3);CHR$(X);"iii"
5114 X=X+4:C=4
5115 RETURN
5120 PRINT E1$;CHR$(Y);CHR$(X);"i";E9$;CHR$(95);E10$;" ";E9$;"r";E10$;
"i"
5121 PRINT E1$;CHR$(Y+1);CHR$(X);"i";CHR$(95);"iri"
5122 PRINT E1$;CHR$(Y+2);CHR$(X);"i   i"
5123 PRINT E1$;CHR$(Y+3);CHR$(X);"i   i"
5124 X=X+6:C=6
5125 RETURN
5130 PRINT E1$;CHR$(Y);CHR$(X);E9$;CHR$(95);E10$;" i"                 'N
5131 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;CHR$(95);"E10$;"i"
5132 PRINT E1$;CHR$(Y+2);CHR$(X);"i";CHR$(95);"i"
5133 PRINT E1$;CHR$(Y+3);CHR$(X);"i ";CHR$(95)
5134 X=X+4:C=4
5135 RETURN
5140 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$    'O
5141 PRINT E1$;CHR$(Y+1);CHR$(X);"i i"
5142 PRINT E1$;CHR$(Y+2);CHR$(X);"i i"
5143 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5144 X=X+4:C=4
5145 RETURN
5150 PRINT E1$;CHR$(Y);CHR$(X);"ip";E9$;CHR$(95);E10$                 'P
5151 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"p";E10$;"r"
5152 PRINT E1$;CHR$(Y+2);CHR$(X);"i"
5153 PRINT E1$;CHR$(Y+3);CHR$(X);"i"
5154 X=X+4:C=4
RETURN
5160 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);"E10$ 'Q
5161 PRINT E1$;CHR$(Y+1);CHR$(X);"i i"
5162 PRINT E1$;CHR$(Y+2);CHR$(X);"i i"
5163 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"im"
5164 X=X+5:C=5
5165 RETURN
5170 PRINT E1$;CHR$(Y);CHR$(X);"ip";E9$;CHR$(95);E10$                 'R
5171 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"p";E10$;"r"
5172 PRINT E1$;CHR$(Y+2);CHR$(X);"i";CHR$(95);E9$;CHR$(95);E10$
5173 PRINT E1$;CHR$(Y+3);CHR$(X);"i i"
5174 X=X+4:C=4
5175 RETURN
5180 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$ 'S
5181 PRINT E1$;CHR$(Y+1);CHR$(X);CHR$(95);E9$;CHR$(95);E10$
5182 PRINT E1$;CHR$(Y+2);CHR$(X);" ";CHR$(95);E9$;CHR$(95);E10$
5183 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
5184 X=X+4:C=4
5185 RETURN
5190 PRINT E1$;CHR$(Y);CHR$(X);"iii"                                  'T
5191 PRINT E1$;CHR$(Y+1);CHR$(X);" i"
5192 PRINT E1$;CHR$(Y+2);CHR$(X);" i"
5193 PRINT E1$;CHR$(Y+3);CHR$(X);" i"
5194 X=X+4:C=4
```

```
6022 PRINT E1$;CHR$(Y+2);CHR$(X);" p";E9$;CHR$(95);E10$
6023 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
6024 X=X+4:C=4
6025 RETURN
6030 PRINT E1$;CHR$(Y);CHR$(X);"i i"                                  '4
6031 PRINT E1$;CHR$(Y+1);CHR$(X);"i i"
6032 PRINT E1$;CHR$(Y+2);CHR$(X);"iii"
6033 PRINT E1$;CHR$(Y+3);CHR$(X);"  i"
6034 X=X+4:C=4
6035 RETURN
6040 PRINT E1$;CHR$(Y);CHR$(X);"iii"                                  '5
6041 PRINT E1$;CHR$(Y+1);CHR$(X);"i";E9$;"pp";E10$
6042 PRINT E1$;CHR$(Y+2);CHR$(X);"  i"
6043 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
6044 X=X+4:C=4
6045 RETURN
6050 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$   '6
6051 PRINT E1$;CHR$(Y+1);CHR$(X);"i"
6052 PRINT E1$;CHR$(Y+2);CHR$(X);"ip";E9$;CHR$(95);E10$
6053 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
6054 X=X+4:C=4
RETURN
6060 PRINT E1$;CHR$(Y);CHR$(X);"iii"                                  '7
6061 PRINT E1$;CHR$(Y+1);CHR$(X);" ";E9$;"r";E10$;"r"
6062 PRINT E1$;CHR$(Y+2);CHR$(X);E9$;"r";E10$;"r"
6063 PRINT E1$;CHR$(Y+3);CHR$(X);"i"
6064 X=X+4:C=4
6065 RETURN
6070 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$   '8
6071 PRINT E1$;CHR$(Y+1);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
6072 PRINT E1$;CHR$(Y+2);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$
6073 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
6074 X=X+4:C=4
6075 RETURN
6080 PRINT E1$;CHR$(Y);CHR$(X);E9$;"r";E10$;"p";E9$;CHR$(95);E10$   '9
6081 PRINT E1$;CHR$(Y+1);CHR$(X);CHR$(95);E9$;"p";E10$;"i"
6082 PRINT E1$;CHR$(Y+2);CHR$(X);"  i"
6083 PRINT E1$;CHR$(Y+3);CHR$(X);CHR$(95);E9$;"p";E10$;"r"
6084 X=X+4:C=4
6085 RETURN
```

Patch Page

Pat Swayne
Software Engineer

REDUCDIR Patch (HUG 885-1120)

HUG member Theodore May experienced some difficulty with the program REDUCDIR on HUG disk 885-1120, and sent in a patch to correct the problem. Although I have not encountered his problem with the program, I found that his patch allowed the program to work under HDOS version 1.6 as well as 2.0, if the version check in the program is removed. The following patch installs Mr. May's modification and removes the version check.

```
>PATCH

PATCH Issue #50.06.00

File Name? REDUCDIR

Address? 42204
042204 = 312/303
042205 = 245/^D      (Control-D)
Address? 43332
043332 = 072/0
043333 = 054/0
043334 = 044/52
043335 = 052/50
043336 = 050/44
043337 = 044/21
043340 = 021/11
043341 = 010/0
043342 = 000/31
043343 = 031/66
043344 = 176/0
043345 = 021/^D
Address? ^D

PATCH Issue #50.06.00

File Name? ^D
```

If you would like to make the modification to the source code of the file, first remove the first 7 lines of code starting with the label START. Change the label GOODVER to START. Locate the label NOTSAME and remove the line:

```
    LDA     GROUP           GET LAST GROUP READ
```

This line is a few lines below NOTSAME. Two lines further down, change IOC.CGN to IOC.CSI, and two lines below this, change:

```
    MOV     M,A             MAKE IT CURRENT GROUP
```

to

```
    MVI     M,0             MAKE IT CURRENT GROUP
```

This completes the changes. Re-assemble the program to get the new version of REDUCDIR.

HDOS ASM Patch

HUG Member R. D. Baertschiger found a "bug" in the HDOS Assembler that causes it to abort an assembly with an "Illegal Channel Number" error under certain circumstances when you are generating a cross reference listing. He sent the following patch to correct the problem.

```
>PATCH

PATCH Issue #50.06.00

File Name? ASM
Patch ID? IFOJIC
Prerequisite Code? IFBEIADPGEFFCF
Address? 64210                  (61165 for HDOS 1.6)
064210 = 022/21
064211 = 032/^D
Address? ^D
Patch Check Code? EDKIGHLG      (GEDKMGNJ for HDOS 1.6)
 - - -
Address? ^D
```

This patch alters a routine that moves file names from one area to another so that 17 characters are moved instead of 18. The most characters a file name can have is 17: 4 for the drive designation (such as SY0:), 8 for the file name, 4 for the period and extension, and one delimiter character at the end.

Epson Graphics in Benton Harbor BASIC

In the "Questions and Answers" column in the October, 1982 Issue 33 of REMark, it was stated that by POKEing location 12121 with 255, you could print characters with a value greater than 127 (8-bit characters) in PRINT CHR$(statements in Benton Harbor BASIC. However, this only works if you print only one character per PRINT statement. If you try to print two or more, as in

```
10 PRINT CHR$(178);CHR$(180)
```

the results are unpredictable. Also, you still cannot print a zero (CHR$(0)). This ability is necessary if you want to write a binary file in Benton Harbor BASIC. (In other words, to simulate CIN(in reverse. See "Using Binary Files in Benton Harbor BASIC" in REMark, Issue 41.) The following patch will let you include characters greater than CHR$(127) and CHR$(0) in PRINT statements in Benton Harbor BASIC.

```
>PATCH

PATCH Issue #50.06.00

File Name? BASIC
Patch ID? IFOJIC
Prerequisite Code? IFBEIADPGEFFCF

Address? 57131
057131 = 177/377
057132 = 167/        (hit RETURN)
057133 = 300/311
057134 = 062/^D      (Control-D)
Address? 74326
074326 = 346/207
074327 = 100/362
074330 = 312/340
074331 = 341/74
074332 = 074/52
074333 = 052/157     (for HDOS 1.6, enter 115)
074334 = 157/112
074335 = 112/303
074336 = 303/343
074337 = 344/74
074340 = 074/52
074341 = 052/152     (for HDOS 1.6, enter 110)
074342 = 152/112
074343 = 112/172
074344 = 172/247
074345 = 247/362
074346 = 362/374
```

```
074347 = 374/74
074350 = 074/276
074351 = 276/43
074352 = 043/302
074353 = 302/363
074354 = 363/74
074355 = 074/173
074356 = 173/276
074357 = 276/312
074360 = 312/374
074361 = 374/74
074362 = 074/43
074363 = 043/^D
Address? ^D
Patch Check Code? DJBJIIBG  (for 1.6, enter DIHLIPFK)
```

If your computer uses a Z80 processor (H/Z-89, 90 or upgraded H8), you can use the following shorter patch instead of the above patch. DO NOT ENTER BOTH PATCHES. Use the above ID and Pre- requisite codes.

```
Address? 57131
057131 = 177/377
057132 = 167/
057133 = 300/311
057134 = 062/^D
Address? 74364
074364 = 176/43
074365 = 247/176
074366 = 362/247
074367 = 363/362
074370 = 074/364
074371 = 303/074
074372 = 344/030
074373 = 074/350
074374 = 043/^D
Address? ^D
Patch Check Code? CMJPPFAC
```

Remember, the above patch is for Z80 processors only, and so it will not work on an H/Z100 computer.

After the patch, you can print characters with the 8th bit set (more than 127), but you still cannot use them in concatenation. For example,

```
B$=CHR$(12)+CHR$(45)+CHR$(0)+CHR$(56)+CHR$(137):
                                    PRINT #1,B$
```

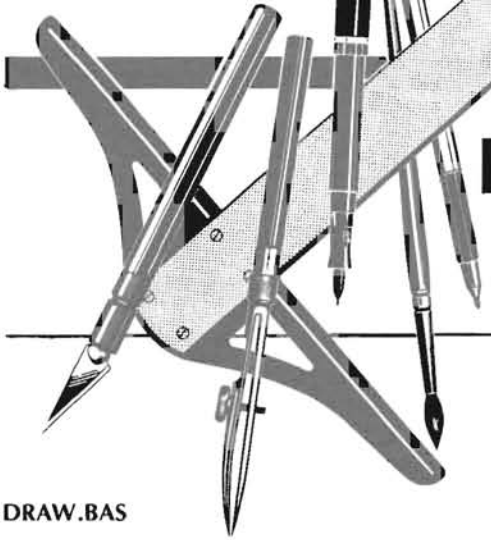
must be restated as:

```
PRINT #1,CHR$(12);CHR$(45);CHR$(0);CHR$(56);CHR$(137)
```

After the patch is made, you cannot use null strings in input statements, as in:

```
10 LINE INPUT "";C$
```

You can, however, type a non printing character (such as Control-E) between the two quote marks to make such input statements work. Because of this problem, you may want to keep an unpatched copy of Benton Harbor BASIC around for programs that do not use 8-bit graphics or do not write binary files.



Draw and Print a Picture in MBASIC

Ron White
RR #1, Box 37
Eldon, Iowa 52554

DRAW.BAS

Here are two short programs which will permit you to draw a picture on the H/Z-89 screen, save it in a file, and retrieve it at a later time.

The programs run under HDOS and MBASIC and are on an H/Z-89. They may run on other hardware and software, but that's for you to decide. I can only vouch for the fact that they work with HDOS and MBASIC.

The first program, DRAW.BAS, permits you to draw a picture using graphics characters, and store these characters (along with their X-Y coordinates) in a text file on the disk. Any other printable character will work as well as graphics.

Line 10 sets the width of the terminal to 255 characters. This is necessary because any setting less than this causes HDOS or MBASIC to insert a line feed, and sometimes a line feed and a carriage return when the width setting is reached, causing errors in the cursor coordinates.

Lines 20 to 130 set up the variables used. The REMarks explain what each line is for.

Line 140 gets the name for the file from the user. Any filename and extension may be used. The name you input will be the name of the file in which the data generated by the program is stored.

Line 150 opens a file for output using the name you supply in line 140.

Line 160 clears the screen and enters the graphics mode.

Line 170 sets the X and Y values for the current cursor position, which is in the upper left corner of the screen. Each time the cursor is moved, the X and Y variables will be incremented or decremented according to which direction the cursor is moved. See lines 240-320.

Line 180 gets the input from the user as to which direction to move the cursor. No carriage return is needed with this input.

The cursor movement keys are the keypad keys 1,2,3,4,6,7,8,9. They move the cursor according to their position in the square they form in the keypad configuration. The regular number keys could be used as well, but the keypad keys are handier.

Lines 190 and 200 will be covered later.

Lines 210 and 220 change your input to a numerical value and check to see if it's a value from 1 to 9. If it isn't, the program goes back to get another input.

Line 230 sends the program to the appropriate line, according to the key you pressed, and these lines (240-320) move the cursor to that position and update the cursor position being stored in the X and Y variables. At the end of these lines the program is returned to get another input from the user.

Line 190. If you have the cursor in a position where you want to print a graphics character (or any other character), press the period key. Line 190 will send the program to line 330, where it will wait for a printing character to be typed.

Line 340 prints that character on the screen.

Line 350 stores the cursor position of that character plus that character itself in the data file that was opened earlier.

Line 360 moves the cursor back 1 position to compensate for its normal movement to the right after printing the character in line 340.

Line 370 sends the program back to 180 to start the process all over.

Line 200. When you are finished with your "picture", press the

DRAW.BAS

```
10 WIDTH 255:REM SET TERMINAL WIDTH TO 255 CHARACTERS
20 E$=CHR$(27):REM ESCAPE
30 CL$=E$+"E":REM CLEAR SCREEN
40 G$=E$+"F":REM GRAPHICS MODE
50 XG$=E$+"G":REM EXIT GRAPHICS MODE
60 C$(2)=E$+"B":REM CURSOR DOWN
70 C$(4)=E$+"D":REM CURSOR LEFT
80 C$(6)=E$+"C":REM CURSOR RIGHT
90 C$(8)=E$+"A":REM CURSOR UP
100 C$(1)=C$(2)+C$(4):REM CURSOR DOWN AND LEFT
110 C$(3)=C$(2)+C$(6):REM CURSOR DOWN AND RIGHT
120 C$(7)=C$(8)+C$(4):REM CURSOR UP AND LEFT
130 C$(9)=C$(8)+C$(6):REM CURSOR UP AND RIGHT
135 REM GET FILE NAME TO STORE DATA
140 INPUT "NAME OF FILE (USE EXTENSION) ";F$
150 OPEN "O",1,F$:REM OPEN FILE FOR OUTPUT
160 PRINT CL$;G$:REM CLEAR SCREEN AND ENTER GRAPHICS MODE
170 X=2:Y=1:REM CURSOR POSITION AT BEGINNING
175 REM GET INPUT AND PROCESS IT
180 I$=INPUT$(1)
190 IF I$="." THEN 330:REM GO GET AND STORE GRAPHICS CHAR AND POSITION
200 IF I$="," THEN 380:REM ALL DONE?
210 N=VAL(I$)
220 IF N < 1 OR N > 9 THEN 180
225 REM PRINT CURSOR MOVE & SET X,Y COORDINATES TO NEW POSITION
230 ON N GOTO 240,250,260,270,280,290,300,310,320
240 PRINT C$(N);:X=X+1:Y=Y-1:GOTO 180
250 PRINT C$(N);:X=X+1:GOTO 180
260 PRINT C$(N);:X=X+1:Y=Y+1:GOTO 180
270 PRINT C$(N);:Y=Y-1:GOTO 180
280 GOTO 180:REM 5 ISN'T A CURSOR POSITION. GO GET ONE
290 PRINT C$(N);:Y=Y+1:GOTO 180
300 PRINT C$(N);:X=X-1:Y=Y-1:GOTO 180
310 PRINT C$(N);:X=X-1:GOTO 180
320 PRINT C$(N);:X=X-1:Y=Y+1:GOTO 180
325 REM GET PRINTING CHARACTER & STORE IT & ITS COORDINATES IN FILE
330 GR$=INPUT$(1)
340 PRINT GR$;
350 PRINT #1,X;Y;GR$
360 PRINT C$(4);:REM SET CURSOR BACK TO SAME POSITION
370 GOTO 180
380 CLOSE:PRINT XG$:END
```

comma key, and line 200 will send the program to line 380, where the file will be closed (a must for sending the last few characters from the buffer to the file) and the graphics mode exited.

As you can see, the only cursor positions saved will be those where characters are printed.

RETRIEVE.BAS

This program will read the file created by DRAW.BAS, and reproduce the "picture" you drew just as you drew it.

Lines 10-60 define the variables used in the program.

Line 80 gets the filename from the user.

Line 90 clears the screen and enters the graphics mode.

Line 100 opens the file named in line 80 for input.

Line 120 checks for the end of the file. When all the data is read, it sends the program to line 160 which closes the file, exits the graphics mode, and ends the program.

Line 130 gets the input from the file. This consists of a pair of X and Y coordinates and a character to be printed at those coordinates.

Line 140 prints the character in the position designated by the X and Y coordinates using direct cursor addressing. This position will be the same place it was when you "drew" it.

Line 150 sends the program back for more data from the file.

This program could be put into another program to print out previously created graphics.

All of the escape sequences are found in either the H/Z-89 manual or the MBASIC manual.

Good "Drawing"

RETRIEVE.BAS

```
10 E$=CHR$(27):REM ESCAPE
20 CL$=E$+"E":REM CLEAR SCREEN
30 G$=E$+"F":REM GRAPHICS MODE
40 XG$=E$+"G":REM EXIT GRAPHICS MODE
50 Y$=E$+"Y":REM DIRECT CURSOR ADDRESSING
60 DEF FN C$(X,Y)=Y$+CHR$(X+31)+CHR$(Y+31):REM CURSOR POSITION
70 REM GET NAME OF FILE WITH STORED DATA
80 INPUT "NAME OF FILE (USE EXTENSION) ";F$
90 PRINT CL$;G$:REM ENTER GRAPHICS MODE AND CLEAR SCREEN
100 OPEN "I",#1,F$:REM OPEN FILE FOR INPUT
110 REM GET DATA
120 IF EOF(1) THEN 160
130 INPUT #1,X,Y,GR$
140 PRINT FN C$(X,Y);GR$
150 GOTO 120
160 CLOSE:PRINT XG$:END
```

About the Author:

Ron White has been in education for the past 22 years. The last 12 as an elementary principal. Ron's hobby interests center on electronics and reading, more specifically, ham radio, computing with the H/Z-89, hi fi, and reading good science fiction books. He states, "All the electronic 'stuff' is HEATHKIT".

Introduction To Z-BASIC Part VII

Gerry Kabelman, C.E.T.
Zenith Data Systems

This is the eighth article in a series of articles dealing with the new commands of the H/Z-100's Z-BASIC over BASIC-80. This month will take a look at the CSRLIN, the DATE$ and TIME$.

The CSRLIN command returns the line (vertical) position of the cursor on the screen to assist in returning to that location at a later time. An example of the use of the CSRLIN is the command shown below.

```
10 CLS:'      CSRLIN.BAS     Version 05.25.83    GK:
20 LOCATE 5,10:PRINT"What is the location where of the cursor"
30 LOCATE 6,10:PRINT"when finished printing this message. ";
40 X=CSRLIN:Y=POS(I)
50 LOCATE 10,10:PRINT"Line"X"and column"Y
60 LOCATE X,Y:LINE INPUT A$
70 END
```

The above program will print the two lines of information and then print the values for X and Y. It then returns to the location of X and Y and waits for an input from the key-board.

The value of X will be from 1 to 25 and using the POS(I) command, as shown, will return a value for Y of 1 to 80. In this example the value for X is 6 and Y is 47.

The CSRLIN is most useful when information is printed on the screen and you don't know how much information was printed, but you need present location of the cursor.

The DATE$ and TIME$ commands are more like variables than commands. When using either DATE$ or TIME$ they may be printed using the PRINT command or used within an expression.

Below is a sample program using the DATE$ and TIME$ commands with the PRINT command and as an expression where the DATE$ and TIME$ are be changed.

```
10'  CLOCK.BAS   Version 05.25.83         GK:
20 CLS:W$="What is the new ":ON ERROR GOTO 110
30 LOCATE 6,20:PRINT"The Date Is: "DATE$
40 LOCATE 8,20:PRINT"The Time Is: "TIME$
50 A$=INKEY$:IF A$<>""THEN 60 ELSE 30
60 LOCATE 12,20:PRINT W$"date? <"DATE$"> ";
70 LINE INPUT A$:IF A$<>""THEN DATE$=A$
80 LOCATE 14,20:PRINT W$"time? <"TIME$"> ";
90 LINE INPUT A$:IF A$<>""THEN TIME$=A$
100 CLS:GOTO 30
110 IF ERR=5 THEN PRINT CHR$(7);:RESUME 100
```

The DATE$ and TIME$ may also be used to create other variables such as to find the year.

```
YR$=RIGHT$(DATE$,4):'          Year
```

The variable YR$ will equal the year, using this year, it is 1983. Also the month, day, hour, minutes, and seconds may be obtained.

```
MN$=LEFT$(DATE$,2):'          Month
DY$=MID$(DATE$,4,2):'         Day
HR$=LEFT$(TIME$,2):'          Hour
MT$=MID$(TIME$,4,2):'         Minutes
SC$=RIGHT$(TIME$,2):'         Seconds
```

Being able to obtain the time and break it apart can be very useful in the timing of a program, such as games or applications that require that elapse time.

That concludes the introduction to Z-BASIC articles, even though not all new commands have been explained. There are several other new commands such as BLOAD, BSAVE, DEF SEG and new options to the OPEN command. However, they are not likely to be used by the beginning programmer.

To help you get started on a project, may I suggest that you try entering the sample program listed below. After you have all the DRAW commands entered correctly, try making other patterns of this type. There are only 49 more to go and I have done one of the hardest.

```
10 CLS:'    MI.BAS   Version 05.27.83     GK:
20 C1=1:C2=2:C3=3:C4=4:C5=5:C6=6:ON ERROR GOTO 290
30 LN=8:FOR I=0 TO LN:CL=4:IF I/2=I\2 THEN CL=5
40 LINE(I*50,1)-(I*50,LN*25),CL
50 LINE(0,I*25)-(LN*50,I*25),CL:NEXT I
:
60 PSET(105,25),C2:DRAW"M125,27D5L10M90,45H3G4f4"
70 DRAW"M96,48R2D2M103,49D2M128,55M152,62"
80 DRAW"M174,56M180,58M214,49M220,51D7M245,53"
90 DRAW"M250,56D6M260,70M257,73M240,72M238,75"
100 DRAW"L4M208,70M194,71M192,74L12M170,76"
110 DRAW"M167,80M163,82M160,85M155,84M160,76"
120 DRAW"M155,75M148,82M144,81M120,94D2M110,97"
130 DRAW"M97,72L30M22,67M10,53M40,45"
140 DRAW"R1M75,37M77,32M80,33M105,25"
150 PSET(233,79),C2:DRAW"M250,82D1r1U1M275,91"
160 DRAW"M280,97M277,100M281,109D10M273,129M268,130"
170 DRAW"M266,135M272,137M283,132M285,128M297,124"
180 DRAW"M305,125M308,128M326,152M323,156M322,162"
190 DRAW"M305,164D5M300,175L8M290,177M293,179"
200 DRAW"M288,188L60U1L72M159,180M165,175M168,168"
210 DRAW"M167,160M165,150M160,138M164,133M163,128"
220 DRAW"M169,112M175,108M180,106M185,102M198,95"
230 DRAW"R2M199,105R5U11M215,86M220,85R2M226,84"
240 DRAW"M224,82L2u1M223,80M229,78M233,79"
250 PAINT(230,90),C2:PAINT(102,38),C2
:
260 LOCATE 5,60:PRINT"Michigan":LOCATE 24,20
270 PRINT"Press any key to continue! ";
280 A$=INPUT$(1):LOCATE 24,1:LIST
:
290 PRINT"Error #"ERR"in line #"ERL:END
```

Have fun using Z-BASIC, creating games and other application programs. Remember, the Heath Users' Group is made up of individuals such as yourself. HUG could use your help in supporting the H/Z-100 computer. Don't keep all those GREAT programs to yourself. Share them with your fellow HUGgies and HUGgers. Also, you may wish to check out the Z-BASIC games (HUG P/N 885-3004-37) for some ideas on how to convert your MBASIC games to Z-BASIC games. Most of the commands discussed in this introduction to Z-BASIC commands were used in one or more of the games.

the one that has it all

H-SCOOP

CONTACT WITH THE FUTURE

THE DOUBLE DENSITY NEWSLETTER

FEATURING
*Specific Headings
*Who to trust
*What's new
*Programming Goodies
*Reports
*Who's doing what
*Classifieds
*Technical issues

Our Trademarks
*Service and Support
*Organization
*Up-to-date
*Fast Service

INFO: H-SCOOP is an independent newsletter rated #1 in several surveys. We support only the H/Z H8, H89 and Z series computers. Large variety of topics and info in every issue.

Rates: One year subscription (12 issues) - $20 for U.S. and Canada sent First Class mail; $27 Overseas, Airmail. Back issues available. Master Charge and Visa accepted. Write or call us at the address below to order or receive more information.

Rated as the BEST newsletter for total H/Z support

TEAM UP WITH THE **WINNERS**

QUIK DATA, INC.

2618 PENN CIRCLE * SHEBOYGAN, WI 53081
(414) 452-4172

We offer full service and support, along with fast turnaround. Not enough space here to publish all prices and items - Call or write for complete catalog or individual item pricing. Check and compare our prices and support and you'll stick with the **WINNERS**! We only carry the best, at the best prices, and with the best support and delivery time!

*Spellbinder; $265 *The Word Plus; $120
*dBase II (Includes Quikmail); $465
*dBPlus; $95
*Tandon Disk Drives:
*100-1; $195 *100-2; $245
*101-4; $325 *848-2; $465
*UltiMeth & LLL Products
*400K and other Drivers-HDOS & CP/M
*ROM's *Software & Mods
*Misc Small Business Software
*Elektrokonsult CP/M Utilities; $29.95 each
*Single 5" drive cabinet; $75
*Dual 5" cabinet; $195 *Dual 8" cabinet; $190
*Cables-many varieties *Diskettes
*NOGDS H89 Color/Sound Board; $399
*Magnolia H89 DD Setup; $495
*Books *HS-XXX Low Cost Software
*Disk format conversion service
*Spike Suppressors; $38 *Rotron Fans; $22
*Vortax speech & music synthesizers
*Electronic print spoolers
*Encon H8/H89 hardware line
*Parallal boards for H8 and H89
*BSR Controller & software

***Much Much more! Call or Write!**

BDOS ERROR!

Bill Simpson
5600 Glenford Street
Los Angeles, CA 90008

While a CP/M BDOS error caused by the failure to reboot a changed disk is generally fatal, resurrection is usually possible. It is certain that your work can be saved should you incur that fatal error while writing or editing an MBASIC program and the same techniques can be applied to many other apparently hopeless situations.

CP/M will not tolerate a disk change unless it is advised of the change by means of a warm boot. Certain information about the files on a disk is placed in memory at the time a disk is booted (warm or cold) and should an attempt be made to write to a different disk, a BDOS error will occur and your system will lock up until a reboot is performed. And, of course, that rebooting drops you out of whatever program you were in and returns you to the operating system with the attendant loss of your entire program (if it was a new one) or all the revisions, updates and corrections (if an old one was being modified).

Fortunately, rebooting does not wipe clean the entire Transient Program Area (TPA) - that portion of memory set aside by CP/M for your use - and under normal circumstances, your program, with all the changes, will still be there somewhere. Your task will be to find it, clean it up a little, move it to a fixed location and then write it to the disk. It's really quite easy using DDT.

The detailed recovery process to be described is known to be accurate only for an H/Z-89 with 64k of memory using CP/M Version 2.2.03 but it seems probable that it will apply to any other system configuration using Heath modified CP/M.

Let us assume that you have finished running one MBASIC program and decide to work on debugging another that resides on a different disk. Since you already have MBASIC in memory, you simply insert the new disk, LOAD the program, RUN it and then proceed to make the necessary modifications. When you're satisfied or otherwise find it time to quit, you triumphantly enter SAVE "MYPROG" only to be presented with Bdos Err On A:. Instantly you realize that you should have RESET when you plugged in the new disk but it's too late now. However, instead of despairing as you once would have, you now know, or will know shortly, how to save the program and all your effort. There is no recourse except to go ahead and tap any key. This will cause a warm boot and produce the CP/M prompt, A>. (Sometimes, for reasons I can't explain, the keyboard will be locked-out and a system reset and cold boot will be necessary. That is quite all right as far as the current problem is concerned and you may fearlessly perform the SHIFT-RESET keypress.)

At this point, there are three important strictures to keep in mind:

1. DO NOT POWER DOWN THE SYSTEM
2. DO NOT RUN ANOTHER PROGRAM
3. DO NOT PROCEED IF YOU ARE TIRED OR YOUR WITS ARE ELSEWHERE (a foul-up here can be permanent). Please wait until you can handle it.

When you are ready, insert a disk containing CP/M's DDT (Dynamic Debugging Tool) and load it (simply enter DDT and a carriage return - do not include a filename on this command line). The DDT sign-on message will appear followed by the DDT prompt. Enter the command: D6200. If all is well (and it should be) and if there are any identifiable ASCII strings, remarks or variables in the first few lines of your old program, you should see them displayed, along with lots of other strange things, in the column to the right of the main hexadecimal display. If there is nothing recognizable to you, then you will simply have to have faith that your program is there beginning at memory location 6207H. The reason it appears so unrecognizable is that it is written in MBASIC's condensed code. The only reassurance that can be offered at this point is that there should be a "b", (62H), at location 6208h. (Compare Figure 1, a sample BASIC program with its stored version, Figure 2.)

The next step is the only one that is not rigorous or is the least bit tricky: you must locate the end of your program. If you have been fortunate enough to end your program with an error handling routine, there will probably be recognizable ASCII strings to signal the end. Otherwise, unless you are very lucky, you will simply have to distinguish the end by the combinations of variables that appear - remember variables are all written out as ASCII strings. The lucky circumstance is that this program ends in

Figure 1

```
100 REM Demo for BDOS.ERR prog
110 FOR J = 0 TO 9
120   FOR I = 0 TO 9
130     PRINT USING "##  ";I+J*10;
140   NEXT I
150   PRINT
160 NEXT J
170 END
```

Figure 2

```
-D6200
6200 61 74 20 61 62 6F 00 24 62 64 00 8F 20 44 65 6D at abo.$bd.. Dem
6210 6F 20 66 6F 72 20 42 44 4F 53 2E 45 52 52 20 70 o for BDOS.ERR p
6220 72 6F 67 00 34 62 6E 00 82 20 4A 20 F0 20 11 20 rog.4bn.. J . .
6230 CE 20 1A 00 46 62 78 00 20 20 82 20 49 20 F0 20 . ..Fbx.  . I .
6240 11 20 CE 20 1A 00 61 62 82 00 20 20 20 20 91 20 . . ..ab..     .
6250 D9 20 22 23 23 20 20 22 3B 49 F2 4A F4 0F 0A 3B . "##  ";I.J...;
6260 00 6B 62 8C 00 20 20 83 20 49 00 73 62 96 00 20 .kb..  . I.sb..
6270 20 91 00 7B 62 A0 00 83 20 4A 00 81 62 AA 00 81  ..{b... J..b...
6280 00 00 00 1A C5 C5 11 30 01 0E 09 CD 05 00 C1 21 .......0.......!
6290 07 00 7E 3D 90 57 1E 00 D5 21 00 02 78 B1 CA 65 ..~=.W...!..x..e
62A0 01 0B 7E 12 13 23 C3 58 01 D1 C1 E5 62 78 B1 CA ..~..#.X....bx..
62B0 87 01 0B 7B E6 07 C2 7A 01 E3 7E 23 E3 6F 7D 17 ...{...z..~#.o}.
```

virgin territory, a section of memory that does not contain remnants of another longer program that was run earlier. Should you be in this fortunate position, the end of your program will be followed by a repeating pattern of hex bytes-probably rows of 00's and FF's which will be represented in the ASCII column as periods. Another clue to seek should the last line of your program have been END, is the number 81 in the hex display portion (81 is the condensed code for END). (Figure 2)

The actual process of searching for the end is to enter repeatedly the DDT command D while watching the ASCII presentation (the right hand column) for some indication of the end of your program. If your program is a long one, you might prefer to use the D,xxxx command where xxxx is the last location (in hex) to be displayed; but be prepared to hit CTRL-S as the data scrolls past or you may miss the end.

Once the end (or the presumed end) has been found, note the hex location of the byte just beyond the last character of your program. We will designate this location as eeee. In our sample program, the last commmand was END and therefore the symbol 81 is the last recognizable entry. However, the three 00 entries following the 81 are also important and should not be lost. (Actually eeee can be any location greater than the end-the important thing is to be sure that the actual end is included.)

To make the presumed end more obvious (an important consideration for a subsequent operation), clean up the locations following the end by using DDT's (F)ill command, e.g. Feeee,pppp,0. This will insert 00 in all memory locations following the end of your program at eeee, up to and including location pppp. This ending point is arbitrary but should be reasonably small to avoid going beyond the end of the TPA. For convenience, I usually go to the end of the current page or to the end of the next page, i.e., to 62FF or 63FF using the current example (Figure 3). Thus, the actual command would be F6283,62FF,0. Of course, should your program have ended in a previously unused part of memory, there is no need to perform this clean-up operation-the end will be obvious.

Now enter the following DDT command: M6206,pppp,100. This is the command to move a copy of the contents of a block of memory extending from hex location 6206 through hex location pppp to a hex location beginning at 100. After this command has been executed, a copy of your program will have been placed in memory starting at the first address of the TPA. In truth, it begins at the second address. You may have noticed that while your program was stored beginning at hex location 6207, we moved a block beginning at 6206h. (See Figure 4)

Figure 3

```
-D6200,62FF
6200 61 74 20 61 62 6F 00 24 62 64 00 8F 20 44 65 6D at abo.$bd.. Dem
6210 6F 20 66 6F 72 20 42 44 4F 53 2E 45 52 52 20 70 o for BDOS.ERR p
6220 72 6F 67 00 34 62 6E 00 82 20 4A 20 F0 20 11 20 rog.4bn.. J . .
6230 CE 20 1A 00 46 62 78 00 20 20 82 20 49 20 F0 20 . ..Fbx.  . I .
6240 11 20 CE 20 1A 00 61 62 82 00 20 20 20 20 91 20 . . ..ab..     .
6250 D9 20 22 23 23 20 20 22 3B 49 F2 4A F4 0F 0A 3B . "## "¦I.J...¦
6260 00 6B 62 8C 00 20 20 83 20 49 00 73 62 96 00 20 .kb..  . I.sb..
6270 20 91 00 7B 62 A0 00 83 20 4A 00 81 62 AA 00 81  ..{b... J..b...
6280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
6290 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

That extra cell at the beginning was needed so that the value FFh could be inserted ahead of the program. This necessary flag is inserted by first entering the DDT command: S100. This will produce the display 100 xx, where xx is whatever happens to be in location 100. Change the contents to FF by simply entering FF and a carriage return. DDT will present the address of the next location, 101, but, since we're through, we terminate the process by entering a period, ".", followed by a carriage return. (Figure 5)

At this point, there should be an FFh in location 100h followed by your program beginning at location 101h (Figure 5). The memory locations beyond your program should be filled with 0's up to the beginning of the next page (more about pages later), i.e., up to the next location ending in 00h — 6300h, 7A00h, AE00h, etc.

It is now necessary to determine how many pages of memory are needed to contain your program. (In a preceding paragraph, it was mentioned that the clean-up process was done for convenience; this is where that convenience is of benefit.) Again use the DDT command D to locate the end of your program in this new location. This is an easy process this time since everything following your program is 00's and you already have a fair idea of the length. Note the line number (leftmost column) of the line containing the end of your program. Round this up to the next hex number ending in 00, i.e., to the next hex multiple of decimal 256 (one page of memory contains 256 locations). Our sample program ended in line 180h; therefore, the number we want is 200h. Other examples would be from 2743 to 2800 or 19B4 to 1A00.

Now subtract 100, the starting location, from the number that you just rounded upward and drop the two rightmost 0's. (Remember, these are hex numbers.) The result will be the number of pages, in hexidecimal, occupied by your program. If, for example, you determined that the next page following your program began at location 2000h, your program requires 1Fh pages of memory-(2000h- 100h)/100h.

You are now through with DDT and can exit with a CTRL-C.

The only thing left to do is to get your program out of memory and on to a disk, the CP/M utility SAVE will do just that. To call up and execute this program requires only a single command line: SAVE nn filename. The first item after SAVE, nn, is the number of pages calculated previously BUT IN DECIMAL. In our previous example, requiring 1Fh pages, we would enter 31 for nn. The filename portion of the SAVE command line includes the drive name, the name you want to use for this saved program and the extension (don't forget the period). Thus your command might be SAVE 31 B:MYPROG.BAS. CP/M will proceed to write out the first 31 pages of memory in the TPA to the disk in drive B and will place MYPROG.BAS in the directory as the name of the saved program. There you will find your newly written program or your old program complete with all the revisions. You will also note, I am certain, that your day has been saved along with your program.

There are, of course, limitations to this process but unless you write very long programs (that will be overwritten by DDT) or improperly identify the end of your program, you should recover your program in its entirety. Whatever you recover, you are certainly ahead of the game. But one word of caution; if you have been modifying an old program and are not sure that you will be able to preserve all of your work or have any doubts about your ability to perform this procedure, save this recovered material under a different name or on a different

disk. There is no point in losing more than your latest effort.

As mentioned, this process is applicable to programs other than MBASIC but the specific location of the stored material may be different and you will have to search for it using DDT. To give a little help, we have found that PIE, the Software Toolworks' text editor, saves things at about location 3480 (there's another copy at B8C0). ED, the CP/M utility editor, puts its material in several locations. In a recent check, edited copy was found at 3 different locations, the first being at 1D4E with the unedited backup at C986. Finally, DDT begins at about location D100 allowing room for a BASIC program approaching 28K in length. Naturally, if your memory is smaller than 64K, the available program space will be reduced accordingly.

Once you understand the above detailed description, all you need to remember, cut-out, or copy are the following 5 steps:

1. Feeee,pppp,0<CR>
   (clean up final page)
2. Mbbbb,eeee,100<CR>
   (move program to location 100)
3. S100<CR>FF<CR>.<CR>
   (change first byte to FF)
4. <CR> (exit DDT)
5. SAVE nn <filename> (save it)

bbbb = start of program (6206 for MBASIC),
eeee = end of program and 3 terminal 00's,
pppp = end of page,
nn = number of pages

One final suggestion: I have saved myself a great amount of time and anxiety in having to execute the above outlined process by adding a few lines of program at the beginning of each of my BASIC programs as they are being developed. These are:

```
1. RESET
2. F$="myprog"
3. INPUT ZQQ$:IF ZQQ$<>"" THEN PRINT F$:END
```

Of course, only the first line serves the avowed purpose but the next line lets me SAVE F$ (without the necessity of quotes) and the INPUT line makes me think about what I'm doing in case I want to protect the program or save it as ASCII. If you are concerned about wearing out your disk with repeated RESETs, it is simple enough to use RUN 100 (or whatever) but that little extra operation is sufficient to remind you to RUN or RESET when the time comes to SAVE your program.

Figure 4

```
-D100
0100 00 24 62 64 00 8F 20 44 65 6D 6F 20 66 6F 72 20 .$bd.. Demo for
0110 42 44 4F 53 2E 45 52 52 20 70 72 6F 67 00 34 62 BDOS.ERR prog.4b
0120 6E 00 82 20 4A 20 F0 20 11 20 CE 20 1A 00 46 62 n.. J . . . . ..Fb
0130 78 00 20 20 82 20 49 20 F0 20 11 20 CE 20 1A 00 x.  . I . . . . .
0140 61 62 82 00 20 20 20 20 91 20 D9 20 22 23 23 20 ab..    . . "##
0150 20 22 3B 49 F2 4A F4 0F 0A 3B 00 6B 62 8C 00 20  ";I.J...;.kb..
0160 20 83 20 49 00 73 62 96 00 20 20 91 00 7B 62 A0  . I.sb..  ..{b.
0170 00 83 20 4A 00 81 62 AA 00 81 00 00 00 00 00 00 .. J..b........
0180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
01A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
01B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Figure 5

```
-D100
0100 FF 24 62 64 00 8F 20 44 65 6D 6F 20 66 6F 72 20 .$bd.. Demo for
0110 42 44 4F 53 2E 45 52 52 20 70 72 6F 67 00 34 62 BDOS.ERR prog.4b
0120 6E 00 82 20 4A 20 F0 20 11 20 CE 20 1A 00 46 62 n.. J . . . . ..Fb
0130 78 00 20 20 82 20 49 20 F0 20 11 20 CE 20 1A 00 x.  . I . . . . .
0140 61 62 82 00 20 20 20 20 91 20 D9 20 22 23 23 20 ab..    . . "##
0150 20 22 3B 49 F2 4A F4 0F 0A 3B 00 6B 62 8C 00 20  ";I.J...;.kb..
0160 20 83 20 49 00 73 62 96 00 20 20 91 00 7B 62 A0  . I.sb..  ..{b.
0170 00 83 20 4A 00 81 62 AA 00 81 00 00 00 00 00 00 .. J..b........
0180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
01A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
01B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Computer Aided Instruction, Part 3

Walt Gillespie
REMark Editor

In the first article of this series, I gave you a planning outline and discussed the first section of that outline, Analysis of your project. In the second article, discussion was centered on sections 2 and 3 of the outline, Design Task, and Choosing a Language. In this the final article on "Computer Aided Instruction", I will cover the last two sections of the outline, Keeping a Clean House and Fine Tuning.

To help in keeping a clean house, one should set up a few basic rules at the very beginning, not internal language rules that is, but, personal rules. These rules should be something along the lines of the following list;

1. Always start the program with a Clear Screen command.
2. Start each new frame with a Clear Screen command.
3. Start writing each frame at the same location.
4. Present only questions to be answered in all CAPS or reverse video.
5. Limit graphics to one area of the screen.
6. Etc.

If you make such a list before you start writing your programs, then any time you have a question about screen handling for instance, you could refer to your list.

The handling of user input should always be done in the same manner from one frame to the next. If you chose to use an automatic type function, e.g., Z$=INPUT$(1), keep it consistent. If you should use this automatic function in most all input situations and then find that you have a question that requires a carriage return, prompt for it. Example....Name the capital of New Jersey:<CR>:. If the student is used to having each answer handled automatically, they might become confused when nothing happens on input. Make your input prompts friendly, not hard to use or cumbersome.

Error trapping can become a very important part of a CAI program. If it is implemented properly, and consistently, it will make the program friendlier, easier to use, and generally receive wider acceptance. There are two types of error trapping that are best used. The first is input error trapping and the second is syntax error trapping.

With input error trapping, if the answer that you are looking for, say, falls between 1 and 50, the input variable can be checked, such as,..IF Z<1 or Z>50 GOTO 20 ELSE 100. In the case of a string input, a subroutine could check for various matches, or a DATA statement could be used to compare two variables. Thus in a case where St. Louis, Saint Louis, ST. LOUIS, or SAINT LOUIS, all could be possible answers only a single input need be checked.

"If a question will be asked a second time, usually due to an incorrect response the first time, clear off the old answer before accepting the new one.", says the Minnesota Educational Computing Consortium. And they add, "Allow for null input. A program should allow for null inputs and not let the system print the sometimes confusing system (syntax error) messages. In general, a null input should simply cause the question to be repeated. Sometimes an additional message or hint can be given suggesting the desired response, such as, 'ENTER A NUMBER PLEASE'".

Error checking, or trapping, for such things as improper file names, nonexistent files, etc., are a must. Nothing can be worse than to have a generally well written program bomb because no checking is done to see if a needed data file is present on the disk. Syntax error checking is sometimes hard to implement. But, when done properly, it can make a good program a great one.

Always keep in mind that a lot of people use only one disk drive with their computer system. If your program requires two or three drives to store data files it could limit sales. Don't try to store everything. Some information is best kept as DATA statements within the program while other types of data are best stored on disk. Too much disk access can slow a program down and discourage the student. The student's reaction time will generally set the pace (speed) of program execution. So don't be overly concerned with optimizing the speed of your program, but don't cause it to be slower than need be.

The means of output of test results is another area that must be considered. You can include the option for printer output. This should be only an option though, as not all students will have printers. It's best to store final test results on disk for later evaluation or comparison. By storing the results on disk and allowing for the fact the student might want to take the test later, an on-screen comparison could be made. If a teacher is administrating the tests, he or she might want to be able to compare the results of a number of students. Thus an added feature of your project could be a program to combine and compare various test results.

Now, to quote the MECC, "When authoring and programming are completed, the work has just begun." You should now set about Fine Tuning your programs.

First test run your programs, use your check list to see that each frame is consistent with the rules. Second, check for unexpected or erroneous input. This is where good error checking will pay off. If you have written your programs in modular fashion, and documented them well, it should be easy to go back and make corrections.

Third, a two part test. The first is the Critical User Test. Find someone who knows about your subject and let them have a go at it. Then find someone who knows nothing of your subject and let them try and bomb it. They will! At this point it's time to go back to the beginning of fine tuning and start through again.

With a fair passage through the fine tuning tests, you should run your project materials through one more final check list.

Final Check List (from the MECC):

,....Programmers questions:

☐ Press the RETURN key. The question should be repeated or help given.

☐ Answer the question with an extremely long string of characters. It should process satisfactorily.

☐ Answer the question and then erase (delete or back space) past the beginning of the question. No disruptions should occur.

☐ Answer the question and then press the RETURN key twice, quickly. The next question or frame should not flip past.

☐ Does the program accept ridiculous answers (e.g., claiming to have 4000 sisters)?

☐ Enter an alpha response where a numeric is expected. It should be processed without a confusing REENTER? message.

☐ When a question must be repeated or a message printed, is the frame held stable? It should not scroll.

,....Author's questions:

☐ Is the instruction presented in a logical sequence?

☐ Is the text blocked and spaced for maximum readability?

☐ Does any information disappear before the user has satisfactory time to read it?

☐ Is the type of answer expected by the computer clear to the user?

☐ Is the use of graphics and/or sound appropriate or is the user distracted by it?

☐ Should graphics and/or sound be added to the program to better enhance the presentation?

☐ Is the material being presented at an appropriate reading level?

☐ Does the program contain errors in spelling, grammar, punctuation? Does it use jargon or slang?

This concludes our discussion on Computer Aided Instruction. I would again like to thank the Minnesota Educational Computing Consortium for allowing me to use their booklet, "A Guide To Developing Instructional Software For The APPLE II Microcomputer", as a reference source in the preparation of these articles.

Give HERO Something To Do With His TIME

Pat Swayne
Software Engineer

If your HERO 1 Robot sits around all day with nothing to do with his time, here is a program that will make him announce the time (with the voice accessory) every fifteen minutes. This program is designed so that a task can be called from it as a subroutine so that HERO could not only tell time, but do other jobs such as intruder detection. Just place a JSR to your routine in place of the three NOP's below the comment "PUT OTHER TASKS HERE", or at 10EH in the assembled program, and place your routine after this program. The program can be assembled in two versions, one where the robot sleeps (10 second sleep intervals) while it is not telling you the time, and one where it displays the time on its LED readout.

Listing 1 is the source code for the program, in HUG XMET format. (The XMET cross assembler is available on HUG disks 885-1123 (HDOS) or 885-1229[-37] (CP/M).) If you are using the AVOCET cross assembler, you will have to remove the space before the accumulator designations in instructions that have them. For example, ASL A should be changed to ASLA. Also, you will need to remove the lines containing RROM and RRAM from the listing and add the follow definitions.

```
OUTCH   EQU     0F7C8H
OUTHEX  EQU     0F7B5H
CLRDIS  EQU     0F65BH
REDIS   EQU     0F64EH
MODE    EQU     0EE1H
CLKDAT  EQU     0C300H
CLKCMD  EQU     0C2C0H
```

Listing 1

```
;       RCLOCK - ET-18 VOICE CLOCK PROGRAM

;       THIS PROGRAM HAS A SUBROUTINE THAT ANNOUNCES THE TIME
;       EVERY 15 MINUTES ON THE QUARTER HOUR.  IT IS DESIGNED
;       TO BE PLACED IN A CONTINUALLY EXECUTING LOOP WITH
;       OTHER TASKS.  IT CAN BE ASSEMBLED INTO TWO VERSIONS,
;       "SLEEP", AND "AWAKE".  IN THE SLEEP VERSION, THE
;       SUBROUTINE PUTS THE ROBOT TO SLEEP FOR 10 SECOND
;       INTERVALS AND RETURNS CONTROL TO THE LOOP BETWEEN
;       SLEEP PERIODS.  IN THE AWAKE VERSION, THE SUBROUTINE
;       RETURNS CONTROL TO THE LOOP IMMEDIATELY AFTER DOING
;       ITS JOB.  IN THIS MODE, IT DISPLAYS THE TIME ON THE
;       ROBOT'S LED DISPLAY CONSTANTLY AS WELL AS ANNOUNCING
;       THE TIME EVERY 15 MINUTES.
;
;       BY KURT TESCHENDORF, HEATH CO.
;
;       MODIFIED, AND WAKE MODE ADDED BY P. SWAYNE, HUG  18-MAR-83

SLEEP   EQU     1               ;ASSMEBLE SLEEP VERSION
AWAKE   EQU     0               ;ASSMEBLE AWAKE VERSION
;                               ;0 = FALSE FOR CP/M, 1 = TRUE
;                               ;0 = TRUE FOR HDOS, 1 = FALSE
;       SET ONLY ONE OF SLEEP OR AWAKE TRUE AND THE OTHER FALSE

        RROM                    ;INCLUDE ROM DEFINITIONS
        RRAM                    ;INCLUDE RAM DEFINITIONS
        ORG     0100H

;       TASK LOOP - CALLS CLOCK AND OTHER TASK(S)

RCLK    LDA A   MODE            ;check if in machine mode
        CMP A   #0FFH
        BNE     RCLK1
        CML                     ;in robot mode so change to machine mode
RCLK1   JSR     CLRDIS          ;CLEAR DISPLAY
RCLK2   JSR     ET18T
;       PUT OTHER TASKS HERE
        NOP                     ;JSR TASK HERE
        NOP
        NOP
        BRA     RCLK2

;********
;****ET18T       ET-18 Robot Time Routines
;**

ET18T   DS      0
        IF      AWAKE
        CLR B                   ;GET ZERO IN B
        JSR     RDCLK           ;GET SECONDS ONES DIGIT
        CMP A   OLDSEC          ;SAME AS OLD SECOND?
        BNE     ET18T1          ;NO, ANNOUNCE TIME
        RTS
ET18T1  STA A   OLDSEC          ;SAVE OLD SECOND
```

```
        JSR     REDIS           ;RESET DISPLAY
        LDA B   #5              ;START WITH HOURS 10'S DIGIT
        JSR     RDCLK           ;GET IT
        AND A   #3              ;REMOVE P.M.
        JSR     OUTHEX          ;DISPLAY 10'S HOURS
        LDA B   #4              ;NEXT, HOUR UNITS
        JSR     RDCLK
        JSR     OUTNUM          ;DISPLAY IT WITH PERIOD
        LDA B   #3              ;AND SO ON FOR REST OF TIME
        JSR     RDCLK
        JSR     OUTHEX
        LDA B   #2
        JSR     RDCLK
        JSR     OUTNUM
        LDA B   #1
        JSR     RDCLK
        JSR     OUTHEX
        CLR B
        JSR     RDCLK
        JSR     OUTHEX
        ENDIF
        LDA B   #3              ;get tens of minutes
        JSR     RDCLK
        JSR     MUL10           ;x 10
        PSH A
        LDA B   #2              ;get minute
        JSR     RDCLK
        PUL B
        ABA
        CMP A   #0              ;check if time to announce time
        BEQ     ET18T2
        CMP A   #15
        BEQ     ET18T2
        CMP A   #30
        BEQ     ET18T2
        CMP A   #45
        BEQ     ET18T2
ET18T1A DS      0
        IF      SLEEP
        CRL
        SLEEP   #1              ;SLEEP 10 SECONDS
        CML
        ENDIF
        RTS                     ;return to main pgm to check senses
ET18T2  CMP A   TIMMIN          ;DONE THIS TIME YET?
        BEQ     ET18T1A         ;YES, EXIT
ET18T3  STA A   TIMMIN          ;MARK THIS TIME DONE
        CRL                     ;announce the time
        SPW     SPTHE
        CML
        LDA B   #5              ;get tens of hours
        BSR     RDCLK
        PSH A
        AND A   #00000011B
        BSR     MUL10
        PSH A
        LDA B   #4              ;get unit hours
        BSR     RDCLK
```

```
;**             ENTRY: ACCB = variable of time to read
;*              EXIT:  ACCA = value of variable of time
;*

RDCLK   LDA A   #0A0H           ;send read command
        STA A   CLKDAT
        STA B   CLKCMD          ;tell clock what we want to read
        LDA A   CLKDAT          ;get variable
        RTS

        IF      AWAKE
;********
;****OUTNUM     SAME AS OUTHEX EXCEPT PERIOD PRINTED
;**              ENTRY: ACCA = NUMBER TO DISPLAY
;*               USES:  ACCA, X

OUTNUM  STA A   OUTNU1+1        ;SET UP INDEX
        LDX     #SEGTBL         ;POINT TO SEGMENT TABLE
OUTNU1  LDA A   0,X             ;GET SEGMENT CODE
        JSR     OUTCH           ;SHOW IT ON DISPLAY
        RTS
SEGTBL  DB      11111110B       ;ZERO
        DB      10110000B       ;ONE
        DB      11101101B       ;TWO
        DB      11111001B       ;THREE
        DB      10110011B       ;FOUR
        DB      11011011B       ;FIVE
        DB      11011111B       ;SIX
        DB      11110000B       ;SEVEN
        DB      11111111B       ;EIGHT
        DB      11111011B       ;NINE
        ENDIF

;********
;****NUMSPK     Say a number between 0 and 9
;**              ENTRY: ACCA = number to speek
;*               USES:  X
;*

NUMSPK  PSH A
        ASL A                   ;calculate offset into number table
        STA A   NUMSP1+1
        LDX     #NUMTBL
NUMSP1  LDX     0,X             ;GET ADDRESS OF SPEECH
        CRL
        SPW     0,X             ;speek indexed
        CML
        PUL A
        RTS
NUMTBL  DW      SPN00           ;number table  (0-9)
        DW      SPN01
        DW      SPN02
        DW      SPN03
        DW      SPN04
        DW      SPN05
        DW      SPN06
        DW      SPN07
        DW      SPN08
        DW      SPN09
```

```
        PUL B
        ABA
        STA A   TIMHOR
        JSR     NUMSPK          ;say the hour
        CRL
        LDA A   TIMMIN
        CMP A   #0              ;say O'CLOCK
        BNE     ET18T4
        SPW     SPOCLK
        BRA     ET18TM
ET18T4  CMP A   #15             ;say 'Fifteen'
        BNE     ET18T5
        SPW     SPNO15
        BRA     ET18TM
ET18T5  CMP A   #30             ;say 'Thirty'
        BNE     ET18T6
        SPW     SPNO30
        BRA     ET18TM
ET18T6  CMP A   #45             ;say 'Fourty-five'
        BNE     ET18TM
        SPW     SPNO40
ET18TM  CML
        PUL A
        AND A   #00000100B      ;check for AM or PM
        BNE     ET18TN
        LDA A   #0
        STA A   AMPM
        CRL
        SPW     SPAM
        BRA     ET18TO
ET18TN  LDA A   #1
        STA A   AMPM
        CRL
        SPW     SPPM
ET18TO  DS      0
        IF      SLEEP
        PAUSE   #2*16           ;PAUSE 2 SECONDS
        SLEEP   #1              ;SLEEP 10 SECONDS
        ENDIF
        CML
        RTS                     ;return to main pgm and do tasks

;********
;****MUL10       Multiply ACCA by 10
;**                 ENTRY:  ACCA = number to be multiplied
;*                  EXIT:   ACCA = number times 10
;*                  USES:   ACCA - ACCB
;*

MUL10   ASL A                   ;* 2
        TAB                     ;SAVE RESULT
        ASL A                   ;* 4
        ASL A                   ;* 8
        ABA                     ;PLUS (* 2) = * 10
        RTS

;********
;****RDCLK       Read variable ACCB from clock
```

```
        DW      SPNO10
        DW      SPNO11
        DW      SPNO12

;********
;****COMMON SPEECH PHONENES
;**

SPNO0   DB      [012H,02CH,03AH,026H]   ;Zero
        DB      [03FH,0FFH]

SPNO1   DB      [02DH,032H,031H,0DH]    ;One
        DB      [03FH,0FFH]

SPNO2   DB      [02AH,036H,037H,037H]   ;Two
        DB      [03FH,0FFH]

SPNO3   DB      [039H,02BH,03CH,029H]   ;Three
        DB      [03FH,0FFH]

SPNO4   DB      [01DH,035H,034H,02BH]   ;Four
        DB      [03FH,0FFH]

SPNO40  DB      [01DH,034H,034H,02BH,02AH,029H] ;Fourty
SPNO5   DB      [01DH,015H,00H,029H,0FH]        ;Five
        DB      [03FH,0FFH]

SPNO6   DB      [01FH,0BH,09H,019H,03H,01FH]    ;Six
        DB      [03FH,0FFH]

SPNO7   DB      [01FH,02H,00H,0FH,0AH,0DH]      ;Seven
        DB      [03FH,0FFH]

SPNO8   DB      [05H,05H,029H,02AH]             ;Eight
        DB      [03FH,0FFH]

SPNO9   DB      [0DH,015H,00H,029H,0DH]         ;Nine
        DB      [03FH,0FFH]

SPNO10  DB      [02AH,02H,00H,0DH]              ;Ten
        DB      [03FH,0FFH]

SPNO11  DB      [02H,018H,02H,00H,0FH,0AH,0DH]  ;Eleven
        DB      [03FH,0FFH]
SPNO12  DB      [02AH,02DH,02H,00H,018H,0FH]    ;Twelve
        DB      [03FH,0FFH]

SPNO15  DB      [01DH,0BH,09H,01DH,02AH,03CH,029H,0DH]  ;Fifteen
        DB      [03FH,0FFH]

SPNO30  DB      [039H,03AH,02BH,02AH,029H]      ;Thirty
        DB      [03FH,0FFH]

SPOCLK  DB      [017H,03EH,019H,018H,015H,023H,019H]    ;O'Clock
        DB      [03FH,0FFH]

SPTHE   DB      [038H,032H,023H]                ;The
        DB      [02AH,015H,00H,029H,0CH]        ;Time
```

```
        DB      [01BH,09H,01FH]                 ;Is
        DB      [0DH,015H,023H,037H]            ;Now
        DB      [03FH,0FFH]

SPAM    DB      [06H,021H,029H,03FH,02H,00H,0CH]        ;AM
        DB      [03FH,0FFH]

SPPM    DB      [025H,03CH,021H,03FH,02H,00H,0CH]       ;PM
        DB      [03FH,0FFH]

TIMMIN  DB      01H             ;minutes
TIMHOR  DB      00H             ;hours
AMPM    DB      00H             ;0 = AM  1 = PM
OLDSEC  DB      0               ;OLD SECOND VALUE

        END     RCLK
```

Below are the hex codes for both versions, which you can enter in manually if you do not have a cross assembler.

Robot Clock (Sleep Version)

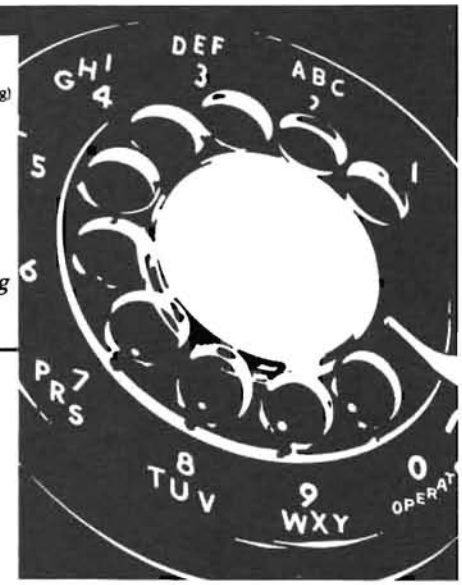
```
0100  B6 0E E1 81 FF 26 01 83
      BD F6 5B BD 01 13 01 01

0110  01 20 F8 C6 03 BD 01 AA
      BD 01 A4 36 C6 02 BD 01

0120  AA 33 1B 81 00 27 12 81
      0F 27 0E 81 1E 27 0A 81

0130  2D 27 06 3F 87 00 01 83
      39 B1 02 7C 27 F5 B7 02

0140  7C 3F 72 02 59 83 C6 05
      8D 60 36 84 03 8D 55 36

0150  C6 04 8D 56 33 1B B7 02
      7D BD 01 B6 3F B6 02 7C

0160  81 00 26 05 72 02 50 20
      19 81 0F 26 05 72 02 3F

0170  20 10 81 1E 26 05 72 02
      49 20 07 81 2D 26 03 72

0180  01 FE 83 32 84 04 26 0B
      86 00 B7 02 7E 3F 72 02

0190  6A 20 09 86 01 B7 02 7E
      3F 72 02 73 8F 00 20 87

01A0  00 01 83 39 48 16 48 48
      1B 39 86 A0 B7 C3 00 F7

01B0  C2 C0 B6 C3 00 39 36 48
      B7 01 BF CE 01 C6 EE 00

01C0  3F 62 00 83 32 39 01 E0
      01 E6 01 EC 01 F2 01 F8

01D0  02 04 02 0B 02 13 02 1B
      02 21 02 28 02 2E 02 37

01E0  12 2C 3A 26 3F FF 2D 32
      31 0D 3F FF 2A 36 37 37

01F0  3F FF 39 2B 3C 29 3F FF
      1D 35 34 2B 3F FF 1D 34

0200  34 2B 2A 29 1D 15 00 29
      0F 3F FF 1F 0B 09 19 03

0210  1F 3F FF 1F 02 00 0F 0A
      0D 3F FF 05 05 29 2A 3F

0220  FF 0D 15 00 29 0D 3F FF
      2A 02 00 0D 3F FF 02 18

0230  02 00 0F 0A 0D 3F FF 2A
      2D 02 00 18 0F 3F FF 1D

0240  0B 09 1D 2A 3C 29 0D 3F
      FF 39 3A 2B 2A 29 3F FF

0250  17 3E 19 18 15 23 19 3F
      FF 38 32 23 2A 15 00 29

0260  0C 1B 09 1F 0D 15 23 37
      3F FF 06 21 29 3F 02 00

0270  0C 3F FF 25 3C 21 3F 02
      00 0C 3F FF 01 00 00 00
```

Robot Clock (Display Version)

```
0100  B6 0E E1 81 FF 26 01 83
      BD F6 5B BD 01 13 01 01

0110  01 20 F8 5F BD 01 E0 B1
      02 CB 26 01 39 B7 02 CB

0120  BD F6 4E C6 05 BD 01 E0
      84 03 BD F7 B5 C6 04 BD

0130  01 E0 BD 01 EC C6 03 BD
      01 E0 BD F7 B5 C6 02 BD

0140  01 E0 BD 01 EC C6 01 BD
      01 E0 BD F7 B5 5F BD 01

0150  E0 BD F7 B5 C6 03 BD 01
      E0 BD 01 DA 36 C6 02 BD

0160  01 E0 33 1B 81 00 27 0D
      81 0F 27 09 81 1E 27 05

0170  81 2D 27 01 39 B1 02 C8
      27 FA B7 02 C8 3F 72 02

0180  A5 83 C6 05 8D 5A 36 84
      03 8D 4F 36 C6 04 8D 50

0190  33 1B B7 02 C9 BD 02 02
      3F B6 02 C8 81 00 26 05

01A0  72 02 9C 20 19 81 0F 26
      05 72 02 8B 20 10 81 1E

01B0  26 05 72 02 95 20 07 81
      2D 26 03 72 02 4A 83 32

01C0  84 04 26 0B 86 00 B7 02
      CA 3F 72 02 B6 20 09 86

01D0  01 B7 02 CA 3F 72 02 BF
      83 39 48 16 48 48 1B 39

01E0  86 A0 B7 C3 00 F7 C2 C0
      B6 C3 00 39 B7 01 F3 CE

01F0  01 F8 A6 00 BD F7 C8 39
      FE B0 ED F9 B3 DB DF F0

0200  FF FB 36 48 B7 02 0B CE
      02 12 EE 00 3F 62 00 83

0210  32 39 02 2C 02 32 02 38
      02 3E 02 44 02 50 02 57

0220  02 5F 02 67 02 6D 02 74
      02 7A 02 83 12 2C 3A 26

0230  3F FF 2D 32 31 0D 3F FF
      2A 36 37 37 3F FF 39 2B

0240  3C 29 3F FF 1D 35 34 2B
      3F FF 1D 34 34 2B 2A 29

0250  1D 15 00 29 0F 3F FF 1F
      0B 09 19 03 1F 3F FF 1F

0260  02 00 0F 0A 0D 3F FF 05
      05 29 2A 3F FF 0D 15 00

0270  29 0D 3F FF 2A 02 00 0D
      3F FF 02 18 02 00 0F 0A

0280  0D 3F FF 2A 2D 02 00 18
      0F 3F FF 1D 0B 09 1D 2A

0290  3C 29 0D 3F FF 39 3A 2B
      2A 29 3F FF 17 3E 19 18

02A0  15 23 19 3F FF 38 32 23
      2A 15 00 29 0C 1B 09 1F

02B0  0D 15 23 37 3F FF 06 21
      29 3F 02 00 0C 3F FF 25

02C0  3C 21 3F 02 00 0C 3F FF
      01 00 00 00
```

Modem Tutorial 83 Part 2

Walt Jung



In the first part of this series, we discussed the basics of what an external RS-232 type modem does, and the fundamentals of the Bell 103 and 212A protocols. With that as a start, we are now ready to get up and running, with a modem and a suitable program to support it. This installment will cover the more detailed considerations in choosing and configuring a modem hardware package for your Heath or Zenith computer.

Pickamodem!

The choice of a modem is the first step towards assembling your computer telecommunication setup, then the appropriate software can be selected to go along with it. Since the last installment covered the basic differences between the direct and acoustic modem types, you should now know how to make a selection based on those differences alone, with the consideration due your budget.

Providing details of program support and operation takes us more into the specific features of individual modem types. While it is not possible in any article of a reasonable length to describe all possible modem features, those that stand out can be hit upon. Examples are the setup requirements for the more simple "dumb" types, and the basics of programming an intelligent modem. Of course, the simple no-frills modems require nothing in the way of actual software programming, so let's take their usage first.

For many people, the nature of usage for a modem might not necessarily ever involve automatic answer, as is typically used within a bulletin board. If you intend to use your modem primarily to call other systems, or personal computer users, and you are willing to do your own dialing and some switch selection, you will likely want to consider some of the currently available direct connect types. Examples of this type of modem are the Anchor Signalman I and the Muraphone MM-100 units.

Both of these units work with the Bell 103 system standard, and offer direct connection capability, as attached to standard modular phone types. The Signalman plugs in between the handset and base, while the Muraphone goes between the wall jack and the phone proper. In use, you simply dial the computer's number on the phone as with any number, and when the answer tone is heard from the modem on the other end, you flip the mode switch to the data position. This will lock the two modems together, and data communications are then ready to begin.

One thing which should be made quite clear at this point is a basic factor, regarding the use of the originate and answer modes. It is almost imperative that you have a modem which is somehow selectable between these modes. Even if you never intend to use it in a bulletin board fashion, sooner or later you will need to use the answer mode. An example would be a link of two personal computers - one needs to be answer, one originate. The Signalman modem automatically switches between originate and answer; with the Muraphone you set it manually. Both units include the phone cables necessary for use, and the Signalman also includes an RS-232C cable (this is purchased separately, with the Muraphone).

For power, the Signalman uses a 9V battery, so it requires no AC line connection. However, since the drain is fairly high, you may want to use either a heavy duty alkaline battery, or the optional booster pack. The Muraphone is powered from the AC line. With either of these units, that is all the hardware, that is peculiar to them, necessary to get you going. There remains a modem program package, which is necessary to support whatever modem you happen to choose. We'll get to the choice of that, later on.

Since the price of the type of modem just described is so attractive (below $100), an obvious question is "Need I pay more?". Well, the answer to that is dependent upon what you want to do, and how you want to use the modem. If you will only be calling other systems, at 300 baud, then the simple modem can be suitable. For $100 to $200 more, you can get a much fancier modem, such as the Hayes Smartmodem (trademark, Hayes Microcomputer Products), which is in the $200 to $300 range. This modem will do virtually everything but format disks for you, but it does require some programming (below), to take advantage of all of the features, particularly for BBS type use. It even talks to you, as well as automatically answering and originating calls, and has LED status indicators. You do of course pay more for this, so ask yourself if that is what you really want (or need).

Be aware that if you want 1200 baud operation as well as 300, you will likely have to pay in the $500 to $700 range, dependent upon features. But, if you are just starting out, it is probably best to stay with one of the more simple units, at least for a while.

In the first part of the series, I used the Smartmodem as an example of a popular modem in the class of those with intelligence. This general group I'll hereafter refer to more in a generic sense, as intelligent modems. The group includes not only the original Smartmodem, but also many of the others patterned after it.

If you do happen to be leaning towards the multi-featured intelligent type modem for Bell 103, 300 baud use, the original Smartmodem is a good choice for applications requiring the features of auto-answer, auto-dial, and software programmability of a number of functions. And, the unit is quite popular with many Heath users. The discussion below covers some of these features, and how they are handled with software. Of course, not everyone will need all of them, and many will be able to get by with less. I'll leave it to you to decide which one of those discussed is your ideal modem. A variety of modems and their suppliers are listed at the end of this article, so you can easily get more information.

RS-232C Intro

Before connecting any modem, some basic familiarity with the RS- 232C standard as it is used with modem communication is in order. What is covered below is by no means exhaustive, but it should suffice, to get you started. If you need more background, see the references at the end.

RS-232C is a serial communications system, and two keys to understanding it are the names used for the devices doing the communicating. As we often see, these devices are referred to as 'DTE' and 'DCE'. The terms DTE (for Data Terminal Equipment) and DCE (for Data Communication Equipment) apply here to the computer (or terminal), and the modem, respectively. RS-232C is for various serial driven devices, such as modems, but obviously it is also used with other devices as well, such as printers.

Figure 2-1 is a simple schematic, which explains the hookup between the DTE device (your computer), and the DCE device (your modem). The DTE device is the frame of reference for the terms Transmitted data and Received data, which use lines 2 and 3, respectively. Line 7 is a common. Note that for communication to work, the DCE device must be receiving on the same pin that the DTE transmits (2). Similarly, the DCE must transmit on the pin the DTE receives on (3). In more simple terms, the computer must talk on the same pin the modem listens, and vice-versa. This hookup will work for the more simple modem types, and may be all you need, if that is what you are going to use.

A more complete usage of the RS-232C function set is shown in the table below, and includes additional lines used for control.

Typically, a modem may use the following signals from the RS-232C set, with those marked '*' corresponding to the minimum set of Figure 2-1.

Function	Pin
Protective ground	(1)
Transmitted data	(2)*
Received data	(3)*
Clear to send	(5)
Data set ready	(6)
Signal ground	(7)*
Received line signal detect (also called carrier detect)	(8)
High speed indicator (on Smartmodem 1200)	(12)
Data terminal ready	(20)
Ring indicator	(22)

The functions of Transmitted Data and Received Data are obvious enough, as described above. Many of the remaining signals become necessary when control functions such as handshaking are to be used, described briefly as follows. Note that you need not be concerned with these lines unless you intend to use the features they allow.

The Data Terminal Ready (DTR) line is used by the computer to signal the modem when to become active. When this line is on, the modem is allowed to answer calls, and conversely, it can be used by the computer to hang up the modem when it is turned off. This is an important function for BBS use, as are the following. The Ring indicator line alternates on and off when a ring is detected. It can be used by the computer if desired, to detect the rings of incoming calls and command the modem to answer the phone by activating the DTR line. The received line signal detect (carrier detect) can be used to sense a loss of carrier from the caller, and reset the BBS system.

It is easy to see that a relatively slow mechanical device such as a printer needs a means of control (handshaking), to signal the computer to stop sending characters. A set of rules which govern just how such a control sequence is carried off is what is called a protocol. Obviously, for any protocol to work, both ends of the link must recognize the same protocol (play the game by the same rules). An example of a basic and familiar protocol is what is known as XON/XOFF protocol; all of us are already familiar with it from using our computers. An XOFF signal (Control-S) will stop transmission of characters, and an XON signal (Control-Q) restarts transmission. Virtually all host telecommunication systems support this simple protocol, allowing the Control-S/Control-Q to be entered either directly from the keyboard, or from software. Modem programs also use more involved protocols, and these will be discussed in a later installment.

If you want further background information on the RS-232C standard, you can either order a copy of the standard from EIA (2001 Eye Street Wash DC 20006), or, see the discussion by Steve Leibson, "The Input/Output Primer, Part 4: The BCD and Serial Interfaces", BYTE, (May 1982, p. 202), as well as "Welcome to the Standards Jungle", by Ian H. Witten, BYTE, (February 1983, p. 136). Also, articles have appeared recently in the >CHUG newsletter, by Mick Topping (April 1983, p. 9) and Dr. William C. Parke (February 1983, p. 5), and will be helpful, being more germane to Heath hardware. You may join CHUG by writing Box 2653, Fairfax VA, 22031; the dues are $12 per year.

Connect That Modem!

To begin the actual hookup, to use any modem you will need an RS- 232C cable to connect your computer's DTE (Data Terminal Equipment) connector to the modem's DCE (Data Communication Equipment) connector. At the modem end, which connector this is should be obvious (the one with two rows of pins, or an umbilical cord, or it may be marked "RS-232C"). At the computer end it may be not quite so obvious, but still it is not hard to decipher. For the H/Z-89, it is the vertical connector marked DTE, nearest the center. On the H8, it is one of 4 (DTE), for the Z-100 it is J2. The cable itself is terminated at either end, with a male and female DB-25 connector pair, and need only be 3-5 feet or so in length. The catalog Heath cable (HCA-11) serves nicely, and has all of the wires you'll ever need. Of course, the modem you select may have a cable with it, there is no universal rule here. If you decide to make up your own, make 1/1 connections between connector pins 1, 2, 3, 4, 5, 6, 7, 8, 12, 20, and 22, from the male to female ends. Use the minimum set if you don't need the handshaking lines.

Number 22 standard hookup wire is fine, as is flat cable.

Of course, just having the wires present in the RS-232C cable and a modem at the other end is simply not all that is required to go on

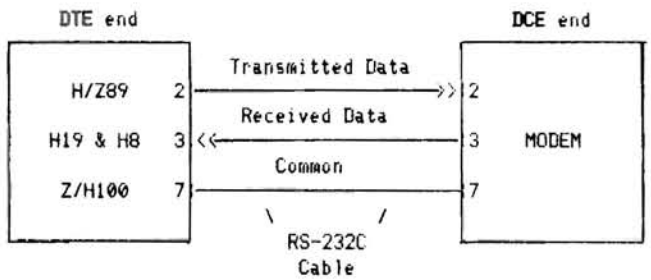


Figure 2-1: Minimum RS-232C cable connections for computer to modem communications. Note **direction** of signals is referred to the **DTE** end.

line. Your Heath or Zenith computer sends and receives serial data via an RS232 port, which in turn is driven by a UART (Universal Asynchronous Receiver Transmitter), either an 8250 for the H/Z-89 and H8, or a 2661 (USART), for the H/Z-100. Fortunately for all of us, Heath has set up things so that there is really little else to be done, for standard usage. If you have the proper serial board operating in your computer, this is virtually the only hardware specific requirement for it. You should make sure that the interrupt jumpers are properly set, to the default position of 5.

Why is it so simple? Well, simply because Heath has seen to it that the operating systems properly set up or initialize all UARTs, for a default standard usage (this includes printer ports, as well, although we are talking here just about modems). In the case of the modem serial interface, this is a baud rate of 300 baud, and the standard port (base) address is at 330Q (D8H) for the H/Z-89 and H8, and at ECH, for the H/Z-100. The default serial data format is initialized to 8 data bits, 1 stop bit, and no parity - a setup which allows all file types, ASCII and binary, to be communicated. The only instance where you need to worry about providing for anything else is, for instance, where you need to use a different baud rate or port address. However, even these are usually taken care of by the modem program (when necessary), so you need not be concerned with them, for now! If in doubt, run your operating system's configuration utility to verify your setup values.

Command	Function
ATDT5551212	Tone dial the number 5551212.
ATDP5551212	Pulse dial the number 5551212.
ATD	Off hook, originate mode, wait for carrier.
ATA	Off hook, make answer mode carrier.
ATO	Return on-line from command state.
ATH	Hang up.
ATZ	Software reset to defaults.
A/	Repeat last command.
+++	Escape from on-line to local command state.

Figure 2-2: Hayes Smartmodem command set (partial). All 'AT' command prefixes are in upper case (only). Commands are terminated by RETURN (with exception of 'A/').

Talking to Modems With Smarts

The Hayes Smartmodem has established a form of relatively simple programming, by the use of ASCII strings for command sequences and dialing. Well, it is no great surprise that this has become a success, since ASCII is a universal "language", independent of hardware and operating systems. In other words, once you have learned to program a Smartmodem, it is the same for HDOS, CP/M- 80, CP/M-85, and/or ZDOS! Speaking further, the commands for both the original and the newer high speed models are basically the same. Since the general Smartmodem command set is supported by other hardware manufacturers, and by software suppliers as well, here is a nutshell description of it.

The Smartmodem has a local command mode to which it defaults, upon power up. From this command state it can be given a variety of commands, dial a number, etc. To command it to a given function, you must always issue the general command prefix 'AT', followed by the remainder of a command. You should note that at this point, we have assumed a working modem program, communicating through the RS-232C serial port to the modem. Figure 2-2 is a capsule summary of the basic commands. From this, you can see that a command to dial using the pulse method (the unit can be either pulse or tone dial) is simply:

ATDP6612175<RETURN>

Within this command, 6612175 is the (local) number you wish to call. For this example, it will get you the BHEC RCPM, if you happen to be in the Baltimore metro area. Sub your own local numbers of course, for your area. Note that the letters 'AT' are in upper case, and also the command line is terminated by hitting your return key (indicated by <RETURN>). If you wanted to call with tone dial, replace the 'P' with a 'T'. To call another area code, you just add it to the number, before the local exchange, with a '1' prefix, as applicable. All AT commands to the Smartmodem must be entered in upper case only.

You can also dial thru a long distance service, such as MCI or Sprint. To do this, you must use the tone dial command, such as:

ATDTXXXXXXX,,,YYYYY,,ZZZZZZZZZZ<RETURN>

Here, the X's are filled in with numbers representing your system's local access number, the Y's correspond to your bill code, while the Z's signify the number you are calling with AC. The commas used insert 2 seconds of delay each, which is necessary to wait for the 2nd dial tone. Vary this, as required to suit the timing locally, as the Smartmodem does not listen for this tone (nor does it listen for a conventional dial tone, it is all done with timing delays).

Any Smartmodem command is repeatable by using the general command 'A/'. Thus a busy number can be very simply redialed just by entering 'A/'. This command, unlike the others, does not need to be terminated by a return.

Result Codes

Not only does the Smartmodem accept commands in ASCII form via the RS-232C port, but it also returns (talks back) with specific responses, in the form of either coded digits or words. These are called simply 'result codes', or coded responses as the result of a specific command or action (Note: do not confuse this with normal incoming data; it is distinctly different).

For example, if you give a Smartmodem the 'ATS0=1' command to answer on one ring, it performs the reset and returns the result, which is the worded response 'OK'. This means it has successfully executed the command. A summary of these result codes are shown in Figure 2-3, and as can be noted, the results can be in either words or digits, with word codes as a default. If you obtain a Smartmodem or another modem similarly programmed, you will soon become accustomed to seeing the modem tell you 'CONNECT' when it gets another modem to answer its call, and 'NO CARRIER' when you disconnect (or the line is busy). And, once in a while you'll see 'ERROR', when things don't go quite right! If you chose the Smartmodem 1200, you can also program it to provide the extended result codes, and it will then return 'CONNECT 1200' when you connect at 1200 baud.

Configuration Switches

Behind the snap-off front panel, both Smartmodems have a set of 8 miniature DIP switches, which are used to configure it for different types of use. The functions of these switches are summarized in Figure 2-4.

In this figure, column 1 is the switch, and column 2 is the factory setting, which is useful in the greatest number of applications. The Comment column indicates the function of the switch setting(s), for both states. Note that switches marked with a * are software alterable. The remaining switches (1, 6-8) can only be changed manually.

To use either Smartmodem to originate calls, you would likely want to use the factory settings, just as shown. Should you want to use a Smartmodem in a BBS type of use, you would want to enable DTR software control (1=UP), as well as Carrier Detect (6=UP), and program the software to provide no echo, no results, as well as monitor/control these lines.

In addition to the above programming considerations, there are also no fewer than 17 internal registers which control variables in the Smartmodem, and they can be either set or interrogated. For example, the S0 register controls on which ring the phone will be answered. To program it to automatically answer on the first ring, the command would be simply ATS0=1.

Conversely, the command ATS0=0 will program the Smartmodem to not answer the phone. The registers are described in great detail in the Hayes manual(s) if you need to alter them, but space does not permit their detailed discussion here. Because of the defaults chosen with the Smartmodem, if you have followed along so far, you have all you need to automatically dial a number and get on line.

A final point of programming consideration, with regard to intelligent modems, is that what has been discussed above is for the two Hayes Smartmodems (see Part I, May 1983). But, you should also be aware that other manufacturers choosing to emulate the command set may or may not do so, in all manner of detail. If you choose to investigate others, be aware of this. An example of another intelligent modem which is similarly programmed is the US Robotics Auto Dial 212A model, shown in Photo 1. The command set of this modem is very close to that of the smart modem, but it does differ somewhat in its manner of configuration (while still being functionally similar).

Photo 1

Digit	Word	Comments
0	OK	Command executed.
1	CONNECT	Carrier detected.
2	RING	Ringing signal detected.
3	NO CARRIER	Carrier lost (or not detected).
4	ERROR	Error in command.
5	CONNECT 1200	Carrier detected at 1200 baud.

Figure 2-3. Smartmodem result codes, in digit (non- verbose) and word (verbose) form.

Switch #	Factory position	Comments
1	DOWN	DOWN forces DTR line always TRUE, with TR light on; UP allows sw control of DTR.
*		
2	UP	UP for worded (verbose) results; DOWN for digit (non-verbose) results.
*		
3	DOWN	DOWN for result codes to be sent; UP for no results (quiet).
*		
4	UP	UP to echo command characters; DOWN for no echo.
*		
5	DOWN	DOWN to not answer a call; UP to answer a call automatically.
6	DOWN	DOWN forces Carrier Detect lead to be TRUE; UP allows computer to detect incoming carrier.
7	UP	UP for single line phones; DOWN for multiple line phones.
8	DOWN	DOWN enables Smartmodem 1200 commands. UP disables command recognition (not used on Smartmodem 300)

Figure 2-4. Configuration switch settings for Hayes Smartmodem and Smartmodem 1200. Those marked * are software programmable.

Summary

So, there you have the hardware basics towards getting up and going with your modem communications setup. I hope that you can see from this that the Smartmodem command set actually allows you to do a great deal, even just from your keyboard. Of course, for the more simple types of use, there are the bare bones units described, which offer you basic performance without all the bells and whistles.

Next time, we will get into the use of BBS systems, and talk about the performance features of various modem software packages. For example, are you familiar with the HDOS and CP/M Heath standbys, CPS? Or Bill Moss's HPLINK, MPLINK, or ZPLINK? If not, you'll learn about them, as well as many others, next time. To whet your appetite, a list of Heath/Zenith BBS systems is provided in Figure 2-5. See you then, and in the meantime, send any questions and comments to the HUG SIG (ID 70001,756), or the BHEC RCPM at (301)-661-2175.

For more information on specific modems, write to the manufacturers below, mentioning this article:

Hayes Microcomputer Products, Inc.
5835 Peachtree Corners East
Norcross, GA 30092

	Smartmodem Smartmodem 1200
US Robotics, Inc. 1123 West Washington Chicago, IL 60607	Auto-Dial 212A
Anchor Automation 16130 Valerio St. Van Nuys, CA 91406	Signalman I
Mura Corporation Westbury, NY 11590	MM-100

Figure 2-5.

List of Heath BBS Systems

Here is a list of all currently known BBS systems, which either run on or otherwise support Heath or Zenith computers, arranged by area code. Please note that systems not clearly designated as "24 hours" usually operate after hours only. Be considerate of their other uses, and do not call them during the week daytime hours.

Information updating or correcting this list will be appreciated, so that future lists can be more complete and detailed. System operators, please send details of your systems, in care of the either REMark or the HUG SIG on CIS, and attempts will be made to republish this list periodically. Thanks!

Phone #	Hours	Location
201-775-8705	24	Ocean, NJ
201-791-3015	24	Fairlawn, NJ
203-674-8915		Avon, CT
206-682-5215	24	Seattle, WA
212-255-7240		New York, NY
213-366-1238	24	San Fernando, CA
213-577-9947		Pasadena, CA
213-749-8442	24	Los Angeles, CA
214-742-1380	24	Dallas, TX
215-288-0262	24	Phila, PA
216-292-7553	24	Cleveland, OH
301-661-2175	24	Balto, MD
301-768-1499	24	Glen Burnie, MD
303-394-2082		Denver, CO
303-632-3019		Colorado Springs, CO
303-985-1108		Denver, CO
305-791-7302		Plantation, FL
305-823-2281		Hialeah, FL
312-852-1305		Downer's Grove, IL
314-291-1854		Bridgeton, MO
314-946-1968		St Charles, MO
404-252-4342		Atlanta, GA
405-848-9329		Oklahoma City, OK
412-824-3565		Pittsburgh, PA
415-365-4915		Redwood City, CA
415-376-6474		San Francisco, CA
415-595-0541		San Carlos, CA
416-231-4174		Toronto, Canada
419-537-1888		Toledo, OH
604-430-8233		Vancouver, Canada
612-778-1213		Cottage Grove, MN
616-982-3682		St. Joseph's, MI
617-237-1511		Wellesley, MA
617-531-9332	24	Peabody, MA
703-360-3812	24	Fairfax, VA
714-629-1943	24	Pomona, CA
714-774-7860	24	Anaheim, CA
716-424-2576		Rochester, NY
716-835-3090		Amherst, NY
801-566-4551	24	Midvale, UT
803-279-5392		Augusta, GA
904-725-4995	24	Jacksonville, FL
907-694-3044		Eagle River, AK
913-362-9583	24	Mission, KS
914-679-8734	24	Woodstock, NY

What Does SET SY: Set?

Bill Tavolga
5151 Windward Ave.
Sarasota, FL 33581

The new hard sector support package (#885-1121) contains an elegant new driver program (HSY.DVD), along with a SETDSK program, that accesses the disk drive constants on track 0. These constants are passed along to any new disks that are initialized, but they are not necessarily the same as those you set with the normal SET.ABS utility. It occurred to me that it would be useful to get a quick look at the constants as they are actually set on the disk in the SY.DVD file. With a little searching in the source code and object code, I found the appropriate bytes. The program that follows will give you a quick readout of these constants, but only for the brand new HUG driver - any other will likely abort the report.

The source code that follows has been heavily annotated, so even someone with a limited experience in Assembly Language should be able to follow it. With disk drives of different densities, brands, etc., I have found this program to be useful. If you are enterprising, you might want to add a write capability to the program, but on your head be it - especially if you don't have a configured backup of SY.DVD handy when there is a crash.

```
* SYSET.ASM              by Wm.N.Tavolga
*                           5151 Windward Ave.
*                           Sarasota  FL  33581
*                                (813) 349-6221
*
*  Program to read and report disk constants as set by SET.ABS program :
*      no. of tracks, sides, seek time, motor one time, head load time
*      on system disk currently in SY0:
*
*   N.B. Program will only work for the latest HSY.DVD version from H.U.G.
*
*    HDOS system calls
*
*
*   H17 ROM routines
*
$DU66   EQU     30106A          (HL)=(BC)/(DE), remainder in DE
$MU10   EQU     30324A          (HL)=(DE)*10
$MU86   EQU     31007A          (HL)=(DE)*A
$TYPTX  EQU     31136A          type line of text that follows
*
SP      EQU     40Q             space
*
        ORG     42200A
SYSET   MVI     A,1             channel 1
        LXI     D,BLOCK         default block
        LXI     H,FNAME         file name to open
        SCALL   .OPENR          open file to read
        JC      ABORT           abort if not possible
SECT5   LXI     B,5             sector number to read
        MVI     A,1             channel 1
        SCALL   .POSIT          set 'cursor' on sector 5
        JC      ABORT           error exit if not possible
READIT  MVI     A,1             open channel 1
        LXI     B,256           read sector
        LXI     D,BUFFER        put it at address of BUFFER
        SCALL   .READ           read it
        JC      ABORT           error exit if no read
REPORT  CALL    $TYPTX                  print headings
        DB      27,'E',12Q,12Q,12Q      clear screen
        DB      9,'Disk constants currently set in SY.DVD : ',12Q
        DB      12Q,9,9,'Tracks',9,'Sides',9,'Seek time (ms)',12Q
        DB      12Q,9,'SY0:',9,240Q
        LXI     H,BUFFER+2      call up byte 3 in buffer
        CALL    SIDSEEK         report side, track code and seek time for SY0:
        CALL    $TYPTX
        DB      12Q,12Q,9,'SY1:',9,240Q
        LXI     H,BUFFER+4      get byte 5
        CALL    SIDSEEK         report side, track code and seek time for SY1:
        CALL    $TYPTX
        DB      12Q,12Q,9,'SY2:',9,240Q
        LXI     H,BUFFER+6      get byte 7
        CALL    SIDSEEK         report side, track code and seek time for SY2:
        CALL    $TYPTX
        DB      12Q,12Q,9,'Motor on (sec) : ',200Q
```

```
        LXI     H,BUFFER+8      get byte 9 - motor on time in sec*2 + 1
        MOV     A,M             move into A
        SUI     1               subtract 1
        RRC                     divide by 2 (rotate right)
        MVI     B,0             put it into BC
        MOV     C,A
        CALL    DOUT            decode & print
        CALL    $TYPTX
        DB      9,'Head load (ms) : ',200Q
        LXI     D,0
        LXI     H,BUFFER+9      get byte 10 - head load in ms/4 + 1
*               note: time value stored is 1/4 of value entered by
*                     SET.ABS program
        MOV     A,M             move data into A
        SUI     1               subtract 1
        MOV     E,A             move data into D,E (multiplicand)
        MVI     D,0
        MVI     A,4             multiplier (4) into A
        CALL    $MU86           subroutine: (HL)=(DE) * A
        MOV     B,H             put into BC
        MOV     C,L
        CALL    DOUT            decode and print
        MVI     A,12Q           print a line feed
        SCALL   .SCOUT
SECT3   LXI     B,3             call up sector 3
        MVI     A,1             channel 1
        SCALL   .POSIT          set sector 'cursor'
        JC      ABORT           quit if not able
READ3   MVI     A,1             channel 1
        LXI     B,256           buffer size
        LXI     D,BUFFER        buffer address
        SCALL   .READ           read the sector
        JC      ABORT           quit if unable
        LXI     H,BUFFER+261Q   find address of byte 177
        MOV     A,M             put data into A
        CPI     0               is it a zero ?
        JZ      NO37            if so then NODKH37 is set
        CALL    $TYPTX             else report that DKH37 is set on
        DB      12Q,9,'DK: is H37 device',212Q
        JMP     QUIT            exit to HDOS
*
NO37    CALL    $TYPTX          report that NODKH37 is set
        DB      12Q,9,'DK: not H37 device',212Q
        JMP     QUIT            exit to HDOS
```

```
        JZ      S801            80 track, 1 side
        CPI     60Q
        JZ      S802            80 track, 2 sides
        JMP     CANT            abort if some other value
*
*   get seek time value and report it
*
SSS     INX     H               incr. to next byte - seek time in ms/2
        MOV     A,M             move it into A
        RLC                     multiply by 2
        MVI     B,0             put it into BC
        MOV     C,A
        JMP     DOUT            decode and print it
*
*   different side & track reports
*
S401    CALL    $TYPTX
        DB      '40',9,SP,SP,'1',9,SP,SP,SP,240Q
        JMP     SSS
S402    CALL    $TYPTX
        DB      '40',9,SP,SP,'2',9,SP,SP,SP,240Q
        JMP     SSS
S801    CALL    $TYPTX
        DB      '80',9,SP,SP,'1',9,SP,SP,SP,240Q
        JMP     SSS
S802    CALL    $TYPTX
        DB      '80',9,SP,SP,'2',9,SP,SP,SP,240Q
        JMP     SSS
*
*  routine to divide, get digit and save remainder
*
DIV     CALL    $DU86           quotient appears in L
        MOV     A,L
        ADI     48              convert to ASCII
        MOV     B,D             remainder in DE, move it into BC
        MOV     C,E                     for next process
        RET
*
*     errors & exits
*
CANT    LXI     H,UNABLE                can't report - prob.wrong SY version
        SCALL   .PRINT
        JMP     QUIT
ABORT   LXI     H,12Q                   unable to open file or read it
```

```
*  routine to convert bytes in BC and save in DIGITS
*
DOUT    LXI     D,1000          load divisor
        CALL    DIV             divide, convert, save remainder
        LXI     H,DIGITS
        MOV     M,A             save digit
*
        LXI     D,100           load divisor
        CALL    DIV             divide, convert, save remainder
        LXI     H,DIGITS+1
        MOV     M,A             save digit
*
        LXI     D,10            load divisor
        CALL    DIV             divide, convert, save remainder
        LXI     H,DIGITS+2
        MOV     M,A             save digit
*
        MOV     A,C             least signif. digit
        ADI     48              convert & save on stack
        LXI     H,DIGITS+3
        MOV     M,A             save digit
*
*     print decimal saved in DIGITS
*
        LXI     H,DIGITS        reload address
        MVI     B,4             load digit counter
DOUTA   MOV     A,M             get a digit - most signif. digit first
        CPI     '0'
        JNZ     DOUTB           skip if leading 0
        DCR     B               decr. counter
        MOV     A,B
        CPI     0               finished ?
        JZ      DOUTB           if yes, then proceed to print
        INX     H                       else go to next address
        JMP     DOUTA                   and get another digit
DOUTB   SCALL   .PRINT          print digits
        RET
*
SIDSEEK MOV     A,M             check byte value
        CPI     0
        JZ      S401            40 track, 1 side
        CPI     20Q
        JZ      S402            40 track, 2 sides
        CPI     40Q
        SCALL   .ERROR
QUIT    XRA     A
        SCALL   .EXIT
*
UNABLE  DB      12Q,7,'Unable to report data',212Q
DIGITS  DS      4,0
        DB      200Q
*
BLOCK   DB      'SY0DVD',0              default block for file
FNAME   DB      'SY',0                  file name to open
BUFFER  DS      256,0                   sector buffer
        END     SYSET
```

*

Update On Heath/Zenith Related Vendors

(These are in addition to vendors listed in the January 1983 Issue

ATG Systems, Inc.
11 Intervale Road
Wellesley Hills, MA 02181

Contact: Jim Jones Phone: 617-431-7870

Comments: Hardware and Software. Consultation Available. Products: ZSpool-Plus features a print queue and print spooling to disk. ZSD-89 provides assignable keys for printing the H19 display, a linefeed or a formfeed. Z80 and CP/M required. Also sell Super19 & Font19 ROMS for the H19.

American Computer
PO Box 386
Haddonfield, NJ 08033

Contact: Rick Schaeffer Phone: 609-939-0802

Comments: Hardware and Software. American Computers discounts products by: Software Toolworks, Sunflower, FBE, TCS, Magnolia, Epson, Okidata, Gemini, Evryware, Hayes, Novation, Zenith Data Systems and many other manufacturers of Zenith Products. American Computers is an Authorized Zenith Data Systems Sales and Service Center. Call or write for a free catalog.

J. E. Brancheau Engineering
Box 67
Trenton, MI 48183

Contact: Jim Brancheau Phone: 313-675-5585

Comments: No hardware or consultation available. Software Products available: HDOS - some common basic programs ($23.00), Practical Basic Programs ($23.00), ELIZA ($18.00), Stock Trends ($24.00), Structured FORTRAN Translator ($24.00), Instant Help ($39.95). CP/M - Instant Help ($39.95). HDOS - Hard Sector 5.25 CP/M - 2.2 or CP/M 85 any format.

C. D. R. Systems Incorporated
7210 Clairemont Mesa Blvd.
San Diego, CA 92111

Contact: Marc Brooks Phone: 619-560-1272

Comments: Hardware, Software and consultation available. Products: FDC-880H Double Density 8" and 5.25" H/Z89, H/Z90 Floppy Disk Controller ($495.00). 10 Megabyte Hard Disk System ($2995.00). 8" Slimline Drive Package for Z100 ($1495.00). See C.D.R. Systems advertisements in this magazine for further information.

CompuMan Services
570 Grace Avenue
PO Box 1416
Panama City, FL 32401

Contact: Sandra Garner Phone: 904-769-1233

Comments: Hardware, Software and consultation available. Products: A variety of software to run on Zenith Systems, leaning to Business Application Programs. Also sell custom designed templates for SpreadSheets. Offers support by installation, training and offers in-house programmers for software modification.

Computer Systems
101 Oneida Avenue
Moorestown, NJ 08057

Contact: Jerry Rubertone Phone: 215-563-1244

Comments: Hardware and Software available. No consultation offered. Products: Hardware, Software discounts to all HUG members on all Zenith Data Systems, Altos, Epson, Okidata, Diablo, Gemini-Star, Daisy Writer, Centronic, Tally, and Nec Printers. Write or call for free catalog and discount price list.

James Czebiniak
199 Valley Road Ext.
Schenectady, NY 12309

Contact: James Czebiniak Phone: 518-869-8723

Comments: Software and consultation available. Product: A Parameter Driven MBASIC Code Generator that provides intelligent screen control and writes complete MBASIC file mgt. code for you, including interactive file maintenance. For 48K H8 and H89: can be modified to suit needs ($50.00).

Extended Technology Systems
1121 Briarwood
Bensalem, PA 19020

Contact: Robert H. Todd, Jr. Phone: 215-752-4604

Comments: Hardware, Software, Firmware and consultation available. Products: Zenith Data Systems dealer, Systems Installed, Hardware and software consultants, Super-19 and Font- 19 Terminal Firmware, Systems software and utilities for HDOS, CP/M and ZDOS. Customization of computers for special user requirements.

Z90/Z100 computers, Printers, many other component systems.

FINA Software
16144 Sunset Blvd. #3
Pacific Palisades, CA 90272

Contact: Larry Fina Phone: 213-454-6393

Comments: Software and consultation available. Hardware available soon. Products: Offers CMDS v2.1, a High Speed Job Command Utility within an enhanced version of SYSCMD.SYS. And, an HDOS Utilities Disk with 12 programs, source available. Send for free documentation.

Fourway Computer Products, Inc.
52578 US 31 North
South Bend, IN 46637

Contact: Donald H. Petersen Phone: 219-277-7720

Comments: Hardware, Software, and consultation available. Products: Zenith computers and terminals, Hewlett-Packard computers and calculators, C-ITOH and Okidata printers, business and technical software. Also provide systems design services and custom software development.

Generic Software
PO Box 790
Marquette, MI 49855

Contact: David J. Powers Phone: 906-475-7151
(call during these hours - 10 a.m. to 5 p.m. EST M-F)

Comments: Software and consultation available. Products: over 20 software products available for H8 and H/Z-89 systems running HDOS or CP/M and for H/Z-100 systems running CP/M 85. Products include data entry utilities database management programs, financial packages, entertainment/educational programs, and home management. Products are available at most Heathkit Stores and from many other Zenith dealers. Call or write for information and FREE catalog. Dealer inquiries invited.

Hilgraeve, Inc.
PO Box 941
Monroe, MI 48161

Contact: Matt Gray Phone: 313-243-0576

Comments: Software and consultation available. Products: ACCESS - a powerful, yet easy to use communications program, provides a flexible link between Z89's or Z100's and any remote computer system.

Horizon Software
PO Box 33066
Philadelphia, PA 19142

Contact: Robert Schild Phone: WRITE

Comments: Software available. Seeking authors of software for the Heath/Zenith 89-90-100 computer systems. Programs of any type on any 5.25" format. High royalties paid. Send programs for immediate evaluation and appraisal. Horizon Software is a retail discounter of high quality software products for Heath/Zenith Computers.

Hoyle and Hoyle Software
716 South Elam Avenue
Greensboro, NC 27403

Contact: Janet C. Hoyle Phone: 919-378-1050

Comments: Software and consultation available. Products: Query! Database Management System totally revised - user suggestions incorporated: Delete key functions, no longer necessary to specify drive names or extensions, easier to create and add, etc. Still ($29.95) Updates ($15.00). Three adventure-type games now in CP/M. A Remarkable Experience ($16.00), A Galactic Experience ($25.00), A Physical Experience ($19.95).

InchSoft 64 Fanchers Street
Pickerington, OH 43147

Contact: Richard E. Lucka Phone: 614-837-8446 (evenings)

Comments: Software and consultation available. Products: EDT - HDOS Full Screen Editor (Requires Z80 CPU in H8 and HDOS 2.0), SAVE - HDOS Disc to Disc File Auto Backup Utility, CNVRT - HDOS Interactive Number System Converter and Calculator.

Lindley Systems
21 Hancock Street
Bedford, MA 01730

Products: DIAMOND Accounting Package, Ultimate Printer drivers work with any known interface, User Programmable Characters for H/Z graphics on MX, NEC, C-ITOH, Okidata dot graphics printers, MAILBAG mailing list software, CHECK MASTER checkbook balancing.

MCA
8 Newfield Lane
Newtown, CT 06470

Contact: John Moran Phone: 203-426-3302

Comments: Software and consultation available. FigForth for H89 under HDOS. Includes a video editor and the Fig-editor, 8080 assembler, two games, and several screens of useful Forth words. $25.00 including documentation. Supports read/write of HDOS files.

MRD Microcomputer Service
10172 Mardel Drive
Cypress, CA 90630

Contact: Wayne Martin Phone: 714-527-4622

Comments: Specialize in Floppy Disk Repair. Hardware Consultation Available. Also sell Floppy Disk Drive and Printers at discount prices.

Magnolia Microsystems, Inc.
2264 Fifteenth Avenue West
Seattle, WA 98119

Contact: Customer Service Dept. Phone 206-285-7266
800-426-2841

Comments: Hardware, Software and consultation available. Products: Floppy, Winchester disk interfaces, subsystems; memory expansions; high performance coprocessor (63K TPA); Local Area Networking; Operating systems; much more, with still more to come. Available thru many Zenith dealers and Heathkit Electronic Centers, nationally and internationally.

MicroArt Corporation
200 Market Bldg./Suite 961
Portland, OR 97201

Contact: Kim Davenport Phone: 1-800-MICROART

Comments: Software available. No consultation offered. Products: IMAGE word/graphics processor offers advanced word processing capability integrated with graphics for the H/Z 67, 89, 90 computers. Printers supported include Epson, Diablo, NEC, C-ITOH, and H/Z25. IMAGE is the only software program for H/Z computers rated all "excellents" by InfoWorld.

Northwest Digital Systems, Inc.
PO Box 15288
Seattle, WA 98115

Contact: Mark Champion Phone: 206-362-6937

Comments: Manufacturers of high resolution (512H by 250V) graphics retrofit for all Heath/Zenith H19 and Z19 video terminals (H89/Z89 not supported). Text features include: Off Screen Memory, Set-up Menu, 80 or 132 columns, 25 or 50 lines, programmable Function Keys, DEC VT-100 Compatible and more. ($849.00)

Jerry A. Phelps
6013 Innes Trace Road
Louisville, KY 40222

Contact: Jerry Phelps Phone: 502-425-4765 (evenings)

Comments: Arcade quality game, LADDERS: Improved version: runs faster, more graphics, cartoons, hazards. Available for CP/M 2.2 (40K) or HDOS 1.6 and up (32K). Requires H/Z89 or H/Z19, H17 and H8 with H8-4 board. See Buss #52 and #65 or H-SCOOP #29 and #36. Supplied on a 5.25 hard, sssd disk. ($21.50) or return old HDOS version plus ($2.55).

Alison C. Phillips, PE
4012 Thoroughgood Drive
Virginia Beach, VA 23455

Contact: Alison C. Phillips Phone: 804-464-4156

Comments: Software and consultation available. CP/M operation and MBASIC-80 programming. Have prepared 250 BASIC tutorials which are available on hard-sectored or soft-sectored 5.25 disks. Tutorials cover the complete range of BASIC-80. a diskette of 12 tutorials is prices at ($12.00). Consultation limited to above areas. Written reply to specific questions ($5.00) each.

Powerline Systems
PO Box 97
Lincroft, NJ 07738

Contact: John W. Preusse Phone: 201-842-5751

Comments: Software and consultation available. Products: New for H/Z89, CP/M: JUPITER - Maintains personal data and financial transaction records for over 25,000 individuals. Many special features including built-in full-screen editor for highly convenient data entry, fast operation, storage economy, predefined report, mailing label, and mergeable file output formats. Please see our ad in this issue.

Quick and Dirty Software
4221 Warwick Drive
Anchorage, AK 99504

Contact: William H. DuBay Phone: 907-563-6333

Comments: Software and consultation available. HDOS and MBASIC required: DMS-II data mamagement system, with label maker and data-merge. ($15.00). DOUBLE Entry Ledger system with trial balance, profit and loss ($10.00) DMS-II For records up to 256 characters. With re-formatter, multiple sorts, default data entry, macros.

Ross Custom Electronics
1307 Darlene Way, Suite A12
Boulder City, NV 89005

Contact: J.D. Ross Phone: 702-293-7426

Comments: Hardware, Software and consultation available. Products: EPROM Programmers (DumBurner) and software to load file to EPROM or EPROM to file. Programmers with software from ($149.00). Documentation, Software and PC Board from ($25.00). H89 or H8 with H-8-5; CP/M or HDOS. TRS80 MODIII Software available April '83.

Secured Computer Systems
8575 Knott Avenue
Buena Park, CA 90620

Contact: Ken or Lowell Halbasch Phone: 714-952-3930 or 714-952-3884

Comments: Hardware, Software and consultation available. Products: 16K memory expansion, 2/3rds card, 2 ports serial and 3 ports parallel, Real Time Clock, IEEE 488, Chassis expansion.

Sjulstad Engineering
503 East Fremont
Northfield, MN 55057

Contact: Dr. Michael Leming or Mr. Michael Sjulstad 507-334-2783 Phone: 507-663-3422

Comments: Hardware, Software and consultation available. Products: 16K add-on RAM($59.95), 256K RAM ($699.00), 128K RAM ($469.00) Remote Video Output ($59.95), Alternate Character Generator. We service and repair most microcomputers.

Studio Computers
999 South Adams
Birmingham, MI 48011

Contact: Ray Massa Phone: 313-645-5365

Comments: Hardware, Software and consultation available. Products: A Complete line of Zenith Hardware and Software, as well as over 20 of our own software packages. Fast Service at Discount Prices, In Business since 1978.

Systems Design Network, Inc.
PO Box 31232
Independence, OH 44131

Contact: G. L. Zychowski Phone: 216-447-1319

Comments: Software and consultation available. Products: FORTRAN compatible forms management system, application generator utilizing CRT forms, and a library of software development tools for FORTRAN systems. Customized software development available.

Technical Micro Systems, Inc.
Dept. H
PO Box 7227
Ann Arbor, MI 48107

Contact: Tom Snoblen Phone: 313-994-0784

Comments: Hardware, Software and consultation available. Products: H-1000, an 8086 upgrade board for the H/Z89. Includes: MS-DOS or CP/M 86, 8MHz 8086, 2 or 4MHz Z80 (Software selectable), CP/M 80 and HDOS RAM Disk. H-1000 runs all H/Z89 Software. H19 upgradable. IBM-PC and Z-100 compatible. VISA, MasterCharge accepted.

Todd Enterprises
1121 Briarwood
Bensalem, PA 19020

Contact: Robert H. Todd, Jr. Phone: 215-752-4604

Comments: Software and consultation available. Products: National Distributor for Public Domain Software in Heath Disk Formats. Libraries carried currently include SIG/M, CPMUG, IBM PC/blue, NYACC public domain catalogs also available. Disk formats: 8", 5" - 40 track, HSSS, SSSS, SSDS, 5" - 80 track, HSDS, SSSS, SSDS.

Weitzman Associates
580 N.W. 99th Way
Pembroke Pines, FL 33024

Contact: George F. Weitzman Phone: 305-431-4043

Comments: Hardware available. Software and consultation not offered. Products: Heat sink assemblies for H-8 boards, Wired, tested joy sticks and joystick kits for the HA-8-3 and HA-89-3 color graphics boards. All Joystick connectors use 30 microinch gold flash contacts.

Wideman Computer Consulting
1320 Pepper Villa Drive
El Cajon, CA 92021

Contact: Graham Wideman

Comments: Software available. Hardware and consultation not offered. Products: IBEm utility which allows running many IBM- PC configured programs on a Z100, by emulating PC's text-video calls (interruption) under ZDOS. ($49.95) plus $3.00 shipping and handling. CA residents add tax.

Zeducomp
PO Box 68
Stirling, NJ 07980

Contact: Stephen E. Hesterman Phone: 201-755-2262

Comments: Software and consultation available. Products: ZED, ZED-85: fast full screen text editors for Z89, Z90, H8/Z19 (CP/M) and for Z-100 (CP/M 85), ($35.00) ZSS: Student Scheduler for universities, high schools, ($350.00). Available from many Heath/Zenith dealers or directly from Zeducomp. Visa, MasterCard, Check.

HEATH USERS' GROUP

Current Local HUG Clubs

(NOTE: This listing is of July 1, 1983. If your club is not listed or you are forming a new club and you would like to have it included in out list, please send the proper information to: Heath Users' Group, Attn: Nancy Strunk, Hilltop Road, St. Joseph, MI 49085)

AK, Eagle River
Alaska HUG
P.O. Box 951
Eagle River, AK 99577
907-694-9908 Group Size 20
Contact Person: Ben Sevier

AK, Ft. Greely
COLD HUG
c/o Stan Lockhart PO Box 229
APO Seattle, WA 98733
907-895-3284 Group Size 3
Contact Person: Stan Lockhart
BB, Newsletter in planning stages

AL, Birmingham
BEARHUG (Birmingham HUG)
c/o Jack Goertz PO Box 2625
Birmingham, AL 35202
205-991-5519 Group Size 20
Contact Person: Jack Goertz
1st Wed of each month 7:00pm
At U of A Rm 217 Cudworth Hall

AL, Huntsville
Huntsville AL HUG
Rt 1, Box 427
Lacey's Spring, AL 35754
205-498-2199
Contact Person: Jeff Hamilton
Meet 2nd Thurs at Intercon Research Corp
Jeff's work no. 205-453-2576

AZ, Phoenix
PHUG (Phoenix Heath Users' GP)
c/o Will Summers PO Box 37783
Phoenix, AZ 85069
Group Size 75
Contact Person: Will Summers, President
2nd Tues at 7:00 p.m. at Phoenix HEC
Membership $5 initiation $12/year

AZ, Tucson
SUNHUG (Tucson HUG)
7109 E Broadway
Tucson, AZ 85710
602-885-6773 Group Size 15
Contact Person: Steven Kutoroff, President
Meet even months first Sunday 2:00 pm
Tucson HEC
Meet odd months first Thurs 7:30 pm

CA, Anaheim
ANAHUG (Anaheim HUG)
330 E. Ball Road
Anaheim, CA 92805
213-330-8118 Group Size 103
Contact Person: Bob Chamberlain, Sec.
3rd Thursday 7:30 PM at HEC
BB 714-774-7860

CA, Campbell
San Jose HUG
2350 S Bascom Avenue
Campbell, CA 95008
408-377-8472 Group Size 70
Contact Person: Gerlene York, Sec.
Meet lst and 3rd Wed 7:00 p.m HEC Campbell

CA, El Cerrito
ECHUG (El Cerrito HUG)
6000 Potrero Avenue
El Cerrito, CA 94530
415-236-8870
Contact Person: Alan Biocca
4th Wednesday at HEC

CA, El Monte
ETUG (ET/ETA 3400 Users Group)
11231 Oak Street
El Monte, CA 91731
Group Size 100
Contact Person: Charles Van Dyke
Newsletter 4 times a year

CA, Fresno
FresHUG (Fresno HUG)
4833 East Santa Ana
Fresno, CA 93726
209-291-6258 Group Size 4
Contact Person: Harlen Collins

CA, Glendora
Southern CA H11 Users Group
430 W. Highland Avenue
Redlands, CA 92373
714-886-4766 Group Size 40
Contact Person: Dr. M.J. Di Girolamo
Meets at 625 E. Palm, Glendora, CA

CA, Los Angeles
Los Angeles HUG
P.O. Box 5334
Pasadena, CA 91107

213-792-4763 Group Size 20
Contact Person: Ray Livingston
1st Thursday 7:00 PM at HEC

CA, Los Angeles
LAETUG (Los Angeles ET3400 GP)
2309 S Flower
Los Angeles, CA 90007
213-749-0261
Contact Person: Gilbert Murillo
Other contact Charlie at 213-443-2237
Contact for meeting time and place

CA, Monterey
Naval Pstgrd Sch Hobby Com Clb
Rec. Services Offices
Monterey, CA 93940
408-646-2466 Group Size 65
Contact Person: Tex Moore, President

CA, Pomona
Pomona HUG
1555 N Orange
Pomona, CA 91767
714-985-5303 Group Size 90
Contact Person: Herb Friedman, President
Meet 4th Thursday each month
at 7:30p.m. at HEC
BB 714-629-1943

CA, Redding
Redding Heath Users' Group
Don Talkington 1018 Freda Lane
Redding, CA 96003
916-244-4563
Contact Person: Don Talkington
Meet 2nd Mon at 7:30 pm at Memorial Hosp.
Meet in Biomedical Engineering Office

CA, Redwood City
BAHUG Bay Area HUG
2001 Middlefield Road
Redwood City, CA 94063
415-365-4915 Group Size 219
Contact Person: Bob Bance, Sec.
2nd Tuesday 7:00 PM at HEC

CA, Riverside
Tri-HUG
5705 Via Sotelo
Riverside, CA 92506
714-683-2929 Group Size 20
Contact Person: Kenny Adcock

CA, Sacramento
SHUG (Sacramento HUG)
1860 Fulton Avenue
Sacramento, CA 95825
916-662-7220 Group Size 35
Contact Person: Gloria Stewart, Sec.
Meet 2nd Wed 7:30pm at Sacramento HEC

CA, San Diego
San Diego HUG
12202 Kingford Court
El Cajon, CA 92021
714-561-2540 Group Size 170
Contact Person: Richard Cobb
1st Wednesday 7:00 PM
at Parkway Jr HS La Mesa

CA, Santa Maria
4168 Glenview Drive
Santa Maria, CA 93455
805-937-6938 Group Size 18
Contact Person: Raymond S. Isenson
Meet 1st Mon 7:00 pm at
Vandenburg Air Force Base

CA, Visalia
Visalia HUG
29924 Road 168
Visalia, CA 93291
209-747-3235 Group Size 3
Contact Person: Peter Shkabara
Meeting time and place
not established yet

CA, Woodland Hills
LUVAHUG
22504 Ventura Blvd.
Woodland Hills, CA 91364
213-883-0531 Group Size 40
Contact Person: Paul S. Townsend
2nd Thursday 7:00 PM at HEC

CANADA, Vancouver, BC
Vancouver Island HUG
2022 Douglas St
Victoria, BC CANADA V8T 4L1
604-384-4711
Contact Person: Greg Greene, President
Meet each month at Excalibur Systems LTD
For further info call above number

CO, Colorado Springs
CSHUG (Colorado Springs HUG)
Colorado Springs, CO 80906
303-632-3019 Group Size 25
Contact Person: Richard Evers
Meet last Thurs each month 7:00 pm
Have 24hr BB 303-634-1158

CO, Denver
DENHUG (Denver HUG)
P.O. Box 20422
Denver, CO 80220
303-394-2082 Group Size 120
Contact Person: Alfred K. Carr, Sec./Treas.
BB 303-423-3224 (24 hrs)
2nd Monday 7:00 PM at HEC

CO, Ft. Collins
FT.HUG (Fort HUG)
822 E. County Road 30
Ft. Collins, CO 80525
303-669-4116
Contact Person: Ted Benglen, II
Meet once a month at present

CT, Avon
CONNHUG (Connecticut HUG)
395 W Main Street
Avon, CT 06001
203-589-3824 Group Size 35
Contact Person: Bob Conlon, President
1st Wednesday at 7:00 pm at HEC
BB 203-674-8915

CT, Mystic
MYSTIC ZDS/HUG
14 Holmes Street
Mystic, CT 06355
203-536-6953
Contact Person: Matthew H. Trask
Last Wednesday at 7:00 pm at
14 Holmes, Mystic, CT

FL, Cocoa Beach
Brevard Heath Users' Group
680 Java Road
Cocoa Beach, FL 32931
305-783-6352 Group Size 12
Contact Person: Gene E. Stillman
Meet last Sun of ea. mo. at 7:00 pm
Meet at Patrick AFB, Comet Rec. Ctr.

FL, Fort Myers
SWFHUG (Southwest Florida HUG)
P.O. Box 05-37
Tice, FL 33905
Contact Person: Robert Sloat
Meet 2nd Tues each month 7:30 p.m.

FL, Fort Walton
NWFHUG (NorthWest Florida HUG)
812 Cherokee Road
Eglin AFB, FL 32542
904-651-2108 Group Size 30
Contact Person: George A Repasy, President
Meetings 2nd Wed at DATATEC Inc. 7:00 pm

FL, Jacksonville
JUG (Jacksonville Users Group)
8262 Arlington Expressway
Jacksonville, FL 32211
904-725-4554 Group Size 40
Contact Person: Jerry Leon
Meet lst Wed each month at HEC Jacksonville
BB 904-725-4995 24 hrs

FL, Miami
Miami Amateur Computer Club
4705 W. 16th Avenue
Hialeah, FL 33012
305-823-2280 Group Size 35
Contact Person: Emileo Crespo
Meet 2nd Thurs each month 7:00 pm at HEC
BB 305-823-2281

FL, Orlando
HUG of Central FL Computer Sc.
121 Talmeda Trail
Maitland, FL 32751
805-644-6848 Group Size 11
Contact Person: Joseph Walker, President
4th Wednesday at various locations

FL, Pensacola

221 E. Government
Pensacola, FL 32501
Contact Person: John Causey
Meet 2nd Tue each month 7:00 pm
at above address
Meet at Professional Business Sys.

FL, Tallhassee
Tally HUG
c/o TACS PO Box 6716
Tallahassee, FL 32314
904-562-1412 Group Size 14
Contact Person: Bill Hill
Meet1st Tues each month 7:30
Meet at Alternative Microcomputing

FL, Tampa
Al Lynch HUG
PO BOX 22906
Tampa, FL 33622
Group Size 40
Contact Person: H. Glenn Tanner, Secretary
Meet lst and 3rd Wed 7:30 pm at Tampa HEC
Dues $10 per year

GA, Atlanta
ATHUG (Atlanta HUG)
5285 Roswell Road
Atlanta, GA 30342
404-436-3677 Group Size 50
Contact Person: Leon Trulove
Meet 1st & 3rd Thurs 7:00-9:00 pm
BB 404-252-4342 6:00 pm to 8:00 am

Ga, Augusta
CSRA Computer Club
PO Box 284
Augusta, GA 30903
404-860-2934
Contact Person: Paul Pennington
Meet 4th Monday at 7:30 pm at
Campus Computer Sys
3830 Washington Rd Martinez, GA 30907

GA, Warner Robins
MGHUG

107 Cherokee Forest Trail
Warner Robins, GA 31093
912-923-6962 Group Size 6
Contact Person: Gerald D. Dalldorf
Meet 4th Wed ea. month at various places
2nd contact: John King 912-923-1977

HI, Hilo
BIHUG (Big Island HUG)
P.O. Box 4271
Hilo, HI 96720
808-959-8985 Group Size 10
Contact Person: R.A. Curtis
Meetings at HELCO Conference Room, Hilo
Meet 1st Thurs each month 7:00 pm

HI, Honolulu
HUGH (HUG Hawaii)
1255 Nuuanu Avenue #1405
Honolulu, HI 96817
808-531-8843 Group Size 45
Contact Person: Jim Branchaud, President
3rd Saturday at Mililani,
1st Wednesday at Kalihi

HONG KONG
Compudragon
273 Prince Edward Road
11/C Kowloon, HONG KONG
3-711-8904
Contact Person: K. T. Lee
Club just organizing

IA, Des Moines
DMA HUG (Des Moines Area HUG)
10275 NE 23rd Ave.
Mitchellville, IA 50169
515-266-2382 Group Size 21
Contact Person: Harold Dykens
Meet 3rd Mon ea. month 7:00 pm

IL, Champaign
CCCC (Champaign Cty Comp Club)
412 Dorchester
Mahomet, IL 61835
312-586-5100 Group Size 12
Contact Person: Roger Fraumann

IL, Davis
NI-HUG Northern IL HUG
427 Lockwood Rt 1
Davis, IL 61019
815-248-2241 Group Size
Contact Person: Jim Isenhart
Just starting

IL, Downers Grove
I-HUG (Illinois HUG)
6116 Lane
Downers Grove, IL 60516
312-971-1660 Group Size 25
Contact Person: Len Bateman
3rd Wednesday at various locations

IL, Downers Grove
HUG Metro (Local Chicago)
15 W. 780 Fillmore
Elmhurst, IL 60126
312-985-2381 Group Size 30
Contact Person: Larry Shipinski, President
Meet 2nd Monday of each Month 7:30 pm at HEC

IL, Peoria
CIHUG (Central Illinois HUG)
408 Bess Street
Washington, IL 61571
309-745-8313 Group Size 17
Contact Person: Ronald Morgan, President
3rd Sunday at 3 pm (Jan, Mar, May, Jul, etc.)

IL, Rockford
Blackhawk Bit Burners
325 Beacon Drive
Belvidere, IL 61008
815-544-5206 Group Size 35
Contact Person: Frank D. Dougherty

IL, Springfield
217-753-5795
Contact Person: Bobby Wright
Club just forming

IN, Indianapolis
Indianapolis HUG (IHUG)
1189 Cumberland Avenue
West Lafayette, IN 47906
317-257-4321 Group Size 60
Contact Person: Robert Wild, President
2nd Wednesday 7:15 PM at HEC

KS, Mission
MUG (Mission Users' Group)
6908 West 98th Street
Overland Park, KS 66212
913-649-0879 Group Size 100
Contact Person: Charles L. Bennett
Meet last Sun of month 2:00 pm
at Mission HEC
BB 913-362-9583 and Newsletter

KS, Wichita
Wichita HUG
1909 Siefkin
Wichita, KS 67208
316-681-3456 Group Size 18
Contact Person: David Horwitz
2nd Sunday of ODD months
2:00 PM at E. Pike Bldg.
Corner of Webb and Kellog in Wichita

LA, New Orleans
NOHUG

1900 Veterans Blvd.
Kenner, LA 70062
504-467-6321 Group Size 60
Contact Person: Nathan Gifford
1st Wednesday at 7:30 PM at HEC

MA, Northampton
Hampshire Computer Club
37 Drewson Drive
Northampton, MA 01060
617-584-6227 Group Size 80
Contact Person: George Scheurer
2nd Tuesday 7 PM
at McConnel Hall Smith College
Beginners Group 1st Tuesday

MA, Peabody
HUG North Shore
6 Susan Drive
Saugus, MA 01906
617-233-2941 Group Size 60
Contact Person: Hal Messinger, President
BB 617-531-9332 24 hours
2nd Wednesday Hilltech Bldg Danvers

MA, Pittsfield
BERCHUG (Berkshire County HUG)
73 Waverly Street
Pittsfield, MA 01201
Contact Person: Paul E. Ouellette, Pres.

MA, Wellesley
HUG'EM
165 Worcester Ave
Wellesley, MA 02181
617-237-1510 Group Size 100
Contact Person: Malcolm Partridge, Dir.
3rd Wed 7:00 p.m. at HEC

MD, Baltimore
Baltimore HUG
6106 Marlora Road
Baltimore, MD 21239
301-323-6093 Group Size 70
Contact Person: William Frey
3rd Thursday 7:00 PM at HEC

MI, Ann Arbor
A-SQR-HUG
895 Starwick Drive
Ann ARbor, MI 48105
313-662-0750 Group Size 9-10
Contact Person: L. E. Geisler, Sec.

Meet last Thurs of each month
Mailing sent on time & place ea. mo.

MI, Detroit
Metro Detroit Area HUG
7716 Winona
Allen Park, MI 48101
313-928-7423 Group Size 50
Contact Person: Chuck Dattolo

MI, Kalamazoo
SMHUG (Southwest Michigan HUG)
623 Wildwood Place
Kalamazoo, MI 49008
616-349-3535 Group Size 50
Contact Person: Al Jacobs, Sec./Treas.
4th Saturday 1:00 pm
at Western Michigan University
Moore Hall, Rm 1034, News Letter

MI, Saint Joseph
BLHUG (Blossomland HUG)
P.O. Box 414
Saint Joseph, MI 49085
Group Size 33
Contact Person: Vance Fisher, Chair Person
1st Tuesday 7:00 pm at various locations
Check HEC for place of meeting

MN, St. Paul-Minneapolis
SMUGH
8895 72nd Street
Cottage Grove, MN 55016
612-459-4382 Group Size 100+
Contact Person: Steve Howard, President
Last Monday at 7:00 pm
(Alt. St Paul & Mpls)

MO, St. Louis
SLHUG (St. Louis HUG)
3794 McKelvey Road
Bridgeton, MO 63044
618-259-8113 Group Size 120
Contact Person: Brad Pulaski, Treas.
Meet 2nd Wed 7:30 pm at HEC

NC, Charlotte
HUG Charlotte
2721 Picardy Place
Charlotte, NC 28209
704-374-6997
Contact Person: Jim Simpson

NC, Fayetteville
Cape Fear Computer & HUG
2454 Vandemere Avenue
Fayetteville, NC 28304
919-485-4586 Group Size 11
Contact Person: Jerry Mills, President
Bi-Weekly 2:00pm on Sundays at homes

NC, Hillsborough
HUG-RTP
Rt 3, Box 39A
Hillsborough, NC 27278
919-73-6678
Contact Person: Joe Williams
Meeting place and time unknown

NE, Omaha
OMAHUG (Omaha HUG)
9207 Maple Street
Omaha, NE 68134
402-391-2071 Group Size 200
Contact Person: Chuck Juvenal, Chairman
3rd Sunday 6:30pm at HEC

NETHERLANDS
Dutch Heath Users' Group
NIEUWE KERKHOF 16
9700 PV Groningen, NETHERLANDS
050-180203 Group Size 90
Contact Person: Evert Jan Stokking

NJ, Fairlawn
HUGNJ (HUG of New Jersey)
3507 Broadway
Fairlawn, NJ 07410
201-791-6935 Group Size 85
Contact Person: Mel Beiman
BB 201-791-6936 24 hours
3rd Monday 8:00 pm at HEC

NJ, Ocean
SHUG (South Jersery HUG)
1013 State Hwy 35
Ocean, NJ 07712
201-775-1231 Group Size 71
Contact Person: James J Jones Jr., Sec.
Meet 1st Wed 7:30 pm at Ocean HEC
BB 201-775-8705 24 hours

NM, Albuquerque
Albuquerque HUG
7205 Minuteman NE
Albuquerque, NM 87109
505-821-7393 Group Size 15+
Contact Person: Jim Pomerleau
Meet 1st & 3rd Sun 7:00 pm at members homes

NY, APO New York
BWHUG (Bentwaters HUG)
PSC Box 3703 RAF Bentwaters
APO New York, NY 09755
Contact Person: Sgt. Rodney Jones

NY, Buffalo
BUG (Buffalo Users Group)
223 Clark Road
Kenmore, NY 14223
Group Size 75
Contact Person: Bob Allen
Meet 3rd Sun 1:30 pm at Amherst HEC

NY, Long Island
Jeri-HUG (Jericho HUG)
PO Box 78
Jericho, NY 11753
516-676-5616 Group Size 65
Contact Person: Alan Scott Dodge, Sec./Treas.
Meet 2nd Thurs 8:00 pm Jericho Pub. Library
Monthly newsletter, software library

NY, North White Plains
North White Plains HUG
Elliott Ser Co 720 White Plns Rd
Scarsdale, NY 10583
Group Size 50
Contact Person: Peter Abramson
Meet 2nd Tues ea. mo. 7:30 pm at HEC

NY, Rochester
RHUG (Rochester HUG)
937 Jefferson Road
Rochester, NY 14623
716-773-0193
Contact Person: Joanne Lang, Chairperson
Last Tuesday at 7:00 pm at HEC

NY, Schenectady
Schenectady HUG
c/o T. Budge 715 Sanders St.
Scotia, NY 12302
518-385-5660 Group Size 12
Contact Person: Walter Whipple
Meet 3rd Wed 7:30 pm at above address

OH, Cincinnati
Cincinnati HUG
10133 Springfield Pike
Woodlawn, OH 45215
513-771-8850 Group Size 50
Contact Person: Roger Svoboda
2nd Tuesday 6:30 PM at HEC, $10.00 dues/year
Newsletter I/O Port

OH, Cleveland
NOHUG (Northeastern Ohio HUG)
4705 Tanglewood Place
Lorain, OH 44053
Group Size 40
Contact Person: Art Petkosek
2nd & 4th Thursday 7:00 PM
at Maple Hts. Library

OH, Cleveland
Cleveland HUG
28100 Chagrin Blvd
Cleveland, OH 44122
216-291-1612 Group Size 10
Contact Person: Gerry Ciganko
1st Thurs 7:00 pm at HEC
BB 216-292-7553 24 hours

OH, Columbus
Columbus HUG
2500 Morse Road
Columbus, OH 43229
614-475-7200 Group Size 50
At HEC

OH, Dayton
Wright-Patterson HUG
4110 Spruce Pine Court
Dayton, OH 45424
513-236-4915 Group Size 36
Contact Person: Jim Moore, President
1st Thursday 4 pm at Wright-Patterson AFB

OH, Toledo
THUG (Toledo HUG)
4804 Mt. Airy Road
Sylvania, OH 43560
419-882-3626 Group Size 30
Contact Person: John F. Priebe, President
Last Sunday 8 pm

OK, Oklahoma City
OKC TUGS
c/o Bill Cadwallader PO Box 1171
Lawton, OK 73502
405-848-7593 Group Size 40
Contact Person: Bob Perry
2nd Sunday at 1:00 pm at HEC
BBS 405-848-9329 24 hours

OKINAWA
OKIHUG (Okinawa Users Group)
Box 376, USAFSO
APO San Francisco, CA 96331
Group Size 13
Contact Person: Carl H. Eaton
Meet on Fridays monthly at 7:30 pm
Meeting place varies

PA, Frazer
FUG (Frazer Users Group)
1641 Princess Anne Drive
Lancaster, PA 17601
717-397-3146 Group Size 60
Contact Person: Dave Hendrie, President
1st Saturday 4:00 pm at Frazer HEC

PA, Philadelphia
Philadelphia Heath Users' Group
6318 Roosevelt Blvd
Philadelphia, PA 19149
215-288-0180 Group Size 135
Contact Person: Henry F. Beechhold, Pres.
Meet 2nd Wed. each month 7:00 pm at HEC 8

PA, Pittsburgh
PittsburgHUG
3482 William Penn Highway
Pittsburgh, PA 15235
412-824-3564 Group Size 35
Contact Person: John C. Schultz, President
3rd Thursday 7:00 pm at HEC

RI, Warwick
HUG-'RI' (HUG of Rhode Island)
558 Greenwich Avenue
Warwick, RI 02886
401-738-5150 Group Size 150
Contact Person: Leo Therrin/Dave Haskell
2nd Wednesday 8 pm at HEC

TN, Knoxville
East Tennessee HUG
110 Northshore Dr.
Knoxville, TN 37919
615-588-0281 Group Size 14
Contact Person: Bruce Cliff
Meet 3rd Thursday 7:00 pm
Meet at Productive Programming Inc.

TN, Memphis
Memphis HUG
6874 Kirby Brooks Drive
Memphis, TN 38115
901-362-8860 Group Size 4
Contact Person: Morris Proctor
Meets at National Cotton Council

TN, Nashville
Mi Te HUG (Middle Tenn HUG)
c/o Radio Ser Ctr 116 17th Ave S
Nashville, TN 37203
615-242-0556
Contact Person: Charlie Wolf
Meet 2nd Monday 6:30 pm
at Radio Service Center

TX, Austin
AHUG Austin Heath Users Group
4206 Tamarack Trail
Austin, TX 78759
512-255-0376
Contact Person: George Koehler
Meet once mo. 8:00-10:00 pm
Applied Research Labs
Univ of Tx 10,000 Burnet Rd Austin

TX, Dallas
DFW HUG (Dallas-Fort Worth)
2715 Ross Avenue
Dallas, TX 75201
214-826-4053 Group Size 70
Contact Person: Henry Gardiner, President
1st Thurs. and 15 days later (Wed.)
at 7:30 pm At HEC BB 214-742-1380

TX, Ft. Worth
FWHUG
6825A Green Oakes Road
Ft. Worth, TX 76116
817-737-8823 Group Size 26
Contact Person: Don Murray
Meets 4th Thursday 7:30 ea. month

TX, Houston
HUG-H
7798 Braniff
Houston, TX 77061
713-644-5689 Group Size 75
Contact Person: Tom McCormick, President

TX, San Antonio
SAHUG (San Antonio HUG)
7111 Blanco Road
San Antonio, TX 78216
512-341-8876 Group Size 65
Contact Person: Tom Schneider
First Tuesday at HEC, 7:30 pm

TX, Wichita Falls
NORTEX HUG (N. Texas S. Okla)
2413 Kemp Blvd in Office World
Wichita Falls, TX 76309
817-322-1007 Group Size 24
Contact Person: Alan D. Martin
Meet third Sat 9:00 am at above address

UT, Castle Dale
Castle Mesa Computer Group
670 N. 90 E. Box 123
Castle Dale, UT 84513
801-381-5173 Group Size 10
Contact Person: Doug Sorensen
Meet 3rd Thurs 5:30 p.m above address

UT, Midvale
UHUG (Utah HUG)
58 E. 7200 South
Midvale, UT 84047
801-566-4628 Group Size 75
Contact Person: Don Greene, President
2nd Wednesday 7:00 pm at HEC

VA, Christiansburg
New River Valley HUG
Christiansburg, VA 24073
703-382-4234 Group Size 35
Contact Person: Ted Fleshman
Meet 1st Thurs 7:30 pm
Christiansburg High School

VA, Fairfax
CHUG (Capital HUG)
P.O. Box 2653
Fairfax, VA 22031
703-759-6176 Group Size 600+
Contact Person: Mike Cogswell, President
3rd Monday 7:30 pm at Fairfax High School
Large Software Library (150+ disks)

VA, Richmond
RHUG (Richmond HUG)
4302 Smithdeal Avenue
Richmond, VA 23225
804-231-6759 Group Size 15
Contact Person: Carlos Chafin
Meet 3rd Monday 7:30 pm
at 2049 W. Broad Street

VA, Virginia Beach
THUG (Tidewater HUG)
1055 Independence Blvd.
Virginia Beach, VA 23455
804-460-0997 Group Size 90
Contact Person: John E. Smith, President
1st & 3rd Tuesday at 7:00 pm at HEC

W. GERMANY, Pirmasens
Pirmasens HUG
Box 1131, 270 Signal Co.
APO New York, NY 09189
Group Size 7
Contact Person: Cpt. James L. Ross, Jr.

WA, Bellevue
Pacific Northwest HUG
c/o Jan Johnson PO BOX 993
Bellevue, WA 98009
206-363-3927 Group Size 150
Contact Person: Nathan Hall
Meet 2nd Thurs odd months 6:00 Tukwila HEC
Meet 2nd Mon even months Seattle HEC

WA, Spokane
SPOHUG (Spokane HUG)
S. 3810 Havana
Spokane, WA 99204
509-448-9727 Group Size 18
Contact Person: Charles Ballinger
Newsletter

WA, Vancouver
Portland-Vancouver HUG
516 SE Chkalov Drive
Vancouver, WA 98663
206-254-4441 Group Size 25
Contact Person: Dan Heims
1st Thursday at 7:30 pm at HEC
Portland OR and Vancouver Area

WI, Milwaukee
MHUG Milwaukee Heath Users Gp
5215 W. Fond Du Lac Ave.
Milwaukee, WI 53216
414-352-3346 Group Size 40
Contact Person: Marvin Olson, Treas.
Meet 3rd Saturday 2:00 pm at HEC

WI, Mosinee
CWHUG (Central Wisconsin HUG)
2294 CTH PB
Mosinee, WI 54455

715-693-3429
Contact Person: Edward Ignace Porwit
Club just started
Call or write for information

CANADA, Calgary, Alberta
HUG (Heath Users of Canada)
101 5809 Macleod Trail South
Calgary, Alberta T2H 0J9 CANADA
403-252-2688
Contact Person: Gary Selman

CANADA, Ottawa, Ontario
HUG 'O' (HUG Ottawa)
866 Merivale Road
Ottawa, ONTARIO K1Z 5Z6 CANADA
613-728-3731 Group Size 30
Contact Person: Brian Fultz, President
2nd Wednesday 8:00 pm at HEC

CANADA, Toronto, Ontario
THUG (Toronto HUG)
1480 Dundas Street E.
Mississauga, ONT. CANADA L4X 2R7
416-273-3797 Group Size 25
Contact Person: Bill Smith

CANADA, Vancouver BC
VANHUG (Vancouver HUG)
3058 Kingsway
Vancouver BC, CANADA V5R 5I7
604-576-9842 Group Size 35
Contact Person: Eric Worthy
Last Monday 7:30 pm at HEC
BB 604-430-8233

HOLLAND, Apeldorn
Dutch HUG
Hofstraat 30
7311 KW Apeldorn HOLLAND
Group Size 70
Contact Person: Evert Jan Stokking

PANAMA CANAL
Canal HUG
P.O. Box 1112
APO Miami, FL 34001
84-4094 Group Size 6
Contact Person: Michael Gulick, President
1st Tuesday 7:30 pm Howard AFB

PUERTO RICO, Rosario
PRHUG (Puerto Rico HUG)
Calle La Paz #706, Miramar
Santurce, PR 00907
809-725-1612 Group Size 21
Contact Person: Joseph Gonzalez
Meet 2nd Sunday of odd numbered months

W. GERMANY, Frankfurt
Frankfurt HUG
American Consulate General FRDCO
APO NY, NY 09757
566187 Group Size 3
Contact Person: Carl Lovett

W. GERMANY, Sprendlingen
HUG-Deutschland
Robert-Bosch-Strasse 32-38
D-6072 Dreieich W. GERMANY
06103/3808 Group Size 200
Contact Person: Egon Becker/Lydia Luguet

QUESTIONS & ANSWERS

(EDITOR'S NOTE: If you need answers to specific questions on software or hardware problems that would be beneficial to other users, please drop us a note to, Question & 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 your question in a future issue. Some of the Questions & Answers are contribute by Zenith Data Systems Software Consultation.)

Q. Can I send my printer escape sequences or control characters from Multiplan?

A. Yes. Use the setup field of the Print Options command. To enter a control code, type ↑ (shift 6) followed by the character. For instance ↑ L is a form feed. Because an escape is a control [, enter it as a ↑ [.

Q. Can a program made or developed with HRUN be placed back on an HDOS disk through HRUN or CP/M's PIP for further use and/or development, or must HUG's program CTOH be used?

A. HRUN is a program that runs under the CP/M operating system. Any disk file created by HRUN is actually a CP/M disk file and if it is to be used under HDOS, it must be transferred through a CP/M-to-HDOS utility (such as CTOH). This simply allows HDOS to recognize the newly transferred file as an HDOS file. If the disk file was created while running HRUN, it may have practical use under HDOS provided it uses only routines common to HDOS.

Q. How can I transfer CP/M files from one computer to another using a modem or direct connection?

A. Standard CP/M PIP can only transfer small ASCII files. Small is 16k to 32k depending on the memory size of your computer. Any other transfers would require special software packages. Using PIP on small ASCII files would require the following steps.

The CONFIGUR utility would need to be set up on both computers so that the RDR: and PUN: devices were assigned to the correct port and matching baud rates. The baud rate must be no greater than that supported by the modem or the speed of the computer. Usually 1200 baud is a safe maximum except with 300 baud modems. Experience will show whether your computer can transfer faster than that. If the file is too big or the speed is too fast, there will be characters missing out of the file.

On the computer that is going to receive the file, you would enter the command:

PIP ?=RDR:[B]

where '?' is the name of the receiving file. The previous contents of the file, if any, will be lost. After this is done, on the computer that is going to send the file you would enter:

PIP PUN:=?

where '?' is the name of the sending file.

Q. In a number of assembly language programs, I have noticed the instruction "CODE PIC". About all I know about this instruction is that it stands for 'Position Independent Code'. No mention of this is made in the HDOS manuals, so any explanation would be helpful. When is it used and why?

A. Some HDOS programs are designed such that they must be relocated in memory, e.g. device drivers. The exact location (or new addresses) may vary from system to system. PIC codes provide a means for HDOS to move relocated programs and keep track of the new addresses.

For an explanation of CODE PIC, see "The HDOS Device Drivers Programmer's Guide" by Al Dallas in REMark issue #20 (September 1981), page 8. (Issue #20 is available in REMark Volume 2, P/N 885-4002.) For an illustration of the use of CODE PIC in a program that is not a device driver, see "A Faster Benton Harbor BASIC" by Dahl Metters in issue #39 (April 1983), page 11. Also, refer to "Guide to Setting Up Utilities as Device Drivers" by Charles Horn in Issue #41, page 45.

Q. How can I power-up my H-89 in hexadecimal radix instead of split-octal? Changing the radix to hex after resetting or power-up is a nuisance.

A. The existing H89 MTR-90 ROM will not allow power-up in hexadecimal radix. The ROM is programmed for split-octal. The source code can be modified and a new ROM burned, but this requires special knowledge and equipment.

Q. I get inconsistent sector errors when doing a media test on my Z-37 and Tandon 100-4 drive. What is wrong? Also, I understand that there is a new ROM for the Z-37. What does it correct/improve and does it have any effect on my problem?

A. The higher density drives allow for such narrow tolerance that soft errors will occur. If the errors become consistent, then it is likely that you are having drive problems.

Some Z-37s have a problem booting properly. The new ROM for the Z-37 moves the head a few tracks and then back to the first track to assure a proper boot. If your drives do not have a boot problem the new ROM, is not necessary.

Q. How can I convert my H-19-A terminal (now that Heath Company has quit making the conversion kit) to an H-88-A?

A. You have two possibilities: 1) order the H-88-A Conversion manual, P/N 595-2709, (P/N 595-2803 for H19 to H88) from Heath Parts Department, and from the manual order each part separately. 2) Check into the DG Super-89 board. Contact DG Electronic Developments Co., 700 South Armstrong, Denison, TX 75020.

BUGGIN' HUG

Introducing Heathkit Online Catalog

Dear HUG,

Heath Company is pleased to announce the introduction of the New Heathkit Online Catalog, which is now available on the CompuServe Information Service.

Heathkit, the world's largest manufacturer of electronic products in easy-to-build kit form, offers it's complete line of over 400 major products on this new electronic shop-at-home catalog.

CompuServe is one to the largest public access timeshare databases that may be accessed from anywhere around the country, usually with just a local phone call. All that is needed is a terminal or personal computer, a modem to connect the phone lines, and a Compuserve subscription which is available at any of the nationwide Heathkit Electronic Centers, most computer stores, or through the Heath Users' Group. Besides the Heathkit Online Catalog, CompuServe offers many areas of service, information, news, and electronic communications, along with the Heath Users' Group Bulletin Board where members can exchange ideas, comments, and programs.

The Heathkit Online Catalog, with an easy-to-use menu driven system and 17 major product categories, is ideal for browsing. If you or your family are involved in cars, boats, amateur radio, alternative energy, woodworking, or weather, you'll find your kit in the online catalog. If your interests lean towards the leading edge of technology, you'll enjoy building TV and stereo equipment among the finest in the world, and the latest in robotics and microcomputer technology.

Also available are bargains and feature products. A convenient search routine allows you to locate the Heathkit Electronic Center nearest you. And to make the shop at home service complete, you can place your order online using your Visa, MasterCard, or Heath Revolving Charge.

You'll find the Heathkit Online Catalog under the Personal Computing Service menu, "Shop at Home"; or by typing Go HTH at any ! prompt in the CompuServe Information Service.

Alan Bose

A Hardware Modification To Prevent A 4MHz I/O Bound Hard-Sectored Disk Controller

Dear HUG,

This modification will allow you to use your modified 4 MHz H89 or H89A with your H-88-1 hard-sectored disk controller; correcting the problem, through a hardware modification, of the H-88-1 becoming I/O bound. My special thanks to Mr. Bill Baldridge of Oakland, CA for testing this modification with me. The Z-89-37 soft-sectored disk controller has full compatibility.

A possible reason for the H-88-1 hard-sectored disk controller becoming I/O bound when operating at 4 MHz is the processor chip on the disk controller will not operate at 4 MHz. This modification is accomplished by allowing the disk controller clock to operate at an unswitched 2 MHz by re-routing the clock on the CPU board at the disk controller port P512. By cutting the trace to the P512 at pin 13 and rerouting pin 13 with wire-wrap wire to U502 pin 8, your H-88-1 hard- sectored disk controller will acquire an unswitched 2 MHz.

Robert A. La Pierre
Concord, CA

Dear HUG,

Many people may be using Benton Harbor BASIC programs that use the "INT" (Integer) function. One example is the "Dollar and Cent" routine given by Pat Swayne in RE-Mark Issue 29.

ALL USERS BEWARE! The INT function works fine, most of the time. The following example shows an error of $1.00 out of $521.00. This is certainly not the accuracy one would expect from a computer.

```
LIST
00010 G1=473
00020 M1=23
00030 B1=19.8
```

```
00040 DIM A(2,2)
00050 A(0,0)=5.2
00060 A9=G1 + M1 + B1 + A(0,0)
00070 PRINT A9, INT(A9)
00080 END
*
*RUN
521             520
```

WRONG! The Interger value of 521 IS NOT 520.

A quick call to the Heath factory software fellows verified that the problem stems from Binary math being used internally. This results in round off errors that are negligible most of the time (hopefully).

This potential problem is not limited to Benton Harbor BASIC. If ANY program you are running uses binary instead of BCD math then BEWARE. If you are not sure, find out from the vendor, especially if it is an accounting program. If it uses binary math you may still be safe if the programmer has included special routines to protect against incorrect roundoff.

I used the "Dollar and Cents" in an accounting program I wrote. I have been using the program for more than EIGHT MONTHS without a single problem. Finally the "magic" numbers came up. I discovered the error because I did not use the "Dollar and Cents" routine (which contains the INT function) in the "print-out" routine and did use it for the screen display. You can imagine my "wonderment" when after eight months of perfect operation, the Total on the hardcopy was $1.00 more than that displayed on the screen!

Aren't these electronic marvels a bag of fun??????

Moral of the story??? Just because that box in front of you has the word "COMPUTER" on it, don't assume that it is ALWAYS accurate. Remember, its really just a very dumb box that only does what it is told to do. Computers DO NOT solve problems, PEOPLE do!

Vincent Bush
Route 1, Box 330
Madison Lake, MN 56063

Dear Vincent,

This letter is in answer to your letter about the INT function in Benton Harbor BASIC. Your sample program can be simplified to the following:

```
00010 A=473
00020 B=23
00030 C=19.8
00040 D=5.2
00050 E=A+B+C+D
00060 PRINT E,INT(E)
```

When you run this program, it prints out 521 and 520, just as your program does. The solution to this problem, since it is a round off error, is to round off the number yourself. This can be done in the sample program by adding the following lines:

```
00070 E=INT(E*100+.5)/100
00080 PRINT E,INT(E)
```

Now, when the program is run, it prints

```
521         520
521         521
```

As you can see, the INT function returns the proper number in line 80. Line 70 can be incorporated as a subroutine into any program that must output accurate dollars-and-cents numbers. Use the subroutine to process the result of each addition or subtraction. If you need to round off to something other than two decimal places, you can use the general form of the round off formula:

X=INT(X*(P ^ 10)+.5)/P ^ 10)

Here, X is the number to be rounded off, and P is the number of decimal places required. It may seem strange that the INT function, which caused the problem in the first place, can be used to fix it, but it works!

Patrick Swayne
HUG Software Engineer

Dear Pat,

Inside back cover of #31 was appreciated, so I am passing along another goodie for your HUGgies:

CAT PRINT (Ref: Issue 12, p.19)

Runs on computer H8, H14, H17, H19 with HDOS.

Would you like a hard copy of CAT on the printer so you know what is on the disk?

1. BOOT up as usual.

2. > PIP REM you type this.

3. :P: LP: = /L/S cr. REM your response to PIP. This also works with /LIST or /LIST/S. At this point, your printer is running and listing the CAT. With a felt pen, put your Vol. No. on top of the page.

4. :P: SY0: /RES replace with any disk, and repeat Step 3. Repeat Steps 4 and 3 for all disks, then:

5. CTRL-D

Note: Don't do Step 4 before Step 3 or you get an ERROR! UNKNOWN DEVICE.

Paul A. Bobbin
7305 Pulaski Highway
Baltimore, MD 21237

Dear Walt,

I would like to take this opportunity to express my appreciation for your publication of the article by William G. Bently "The Next Step Is C". I strongly support the fundamental thrust of the article that the use of structured languages improves the reliability and maintainability of our programs. That these languages are usually compilers also increases the performance of our systems. Articles such as this, and those by Henry Fale on Pascal, are most useful in helping your readers obtain more effective use of their computers.

I suspect that there will never be a perfect language for all applications. Depending on the application, I have found C/80 and various implementations of Pascal to be effective tools in getting the job done. After all is said and done, the objective is getting the job done efficiently for the user, the programmer, and lastly the computer - not necessarily which language is used. In this respect the structured languages will prove themselves.

Based on my experience of moving code between various implementations of C and Pascal, I have a difficult time supporting Mr. Bently's contention that C is more portable than Pascal. Portability, however, is but a small factor in the overall situation.

I agree that C tends to be less wordy than Pascal and it supports highly concise statements. I have found these attributes of C tend to make documenting C programs more time consuming than those written in Pascal. More importantly, I have found that people accustomed to thinking in high level terms more rapidly learn Pascal and that people well versed in assembly language tend to gravitate, at least initially, to C. In this respect I strongly agree with Mr. Bently that "the serious student should first become acquainted with assembly language, since many of the basic data types and operations are similiar". The strong type checking of Pascal has been a big help in avoiding those hard to find bugs which I occasionally encountered when working with C and frequently ran into when working with assembly language code.

None of the above should be taken to imply that outstanding programs cannot be written in C or other languages. The programs from the Software Toolworks are excellent examples of this. C and Pascal are both good languages which have high utility in realizing the full potential of our talents and the capabilities of our hardware. COBOL and FORTRAN can also be used to produce outstanding programs however they tend not to

support the structured top-down approach as well as C, PL/1 and Pascal. Naturally, one can also produce pure garbage in any of the available languages.

In looking back over the past few years I find that although some of our club members have used C all of them have converted to Pascal. This movement appears to be due primarily to ease of learning. Pascal was designed to facilitate the learning of the concepts and techniques of computer science and programming. That our area has a considerable military and government component also seems to play a role due to the ease of shifting from Pascal to Ada. I have found it fairly easy, and in some cases trivial, to convert code between Ada and Pascal. This is due to the fact that all four of the software firms which produced Ada language specifications used Pascal as their philosophical foundation.

Although I am clearly biased in favor of Pascal my main recommendation is that each individual look at and evaluate all of the available languages before deciding which one is best suited to his/her environment and applications. In any event, do yourself a favor and don't just look at the first language that comes along.

Fred Pospeschil
3108 Jackson St.
Bellevue, NE 68005

Info Needed for H-10 Paper Tape Punch

Dear HUG,

I am a new member of HUG. I have an H-10 paper tape punch, and cannot find any information on it. I called Heath Technical Assistance, and the person I talked with said he didn't know anything about the H-10. If Heath doesn't know, who does? I am an N. C. Programmer and want to start programming at home. I have an H-89 computer and Okidata 82A printer. In Numerical Control Programming, a punched paper tape is used to load the programs into the lathe, mill, or other such devices or machines, hence the H-10. Maybe you have a better suggestion? I would like a list of the articles that might have to do with the H-10 5Xor N. C. Programming, or information or help on where to get such information. A friend of mine said there was a program (disk) on N. C. Lathes, but said there were other items on the disk, that's fine too. I would appreciate any help you could give.

I have the schematics and manual for the H-10 but I am not electronically enclined, maybe one of the HUG members might have wiring instructions on how the pin to pin wiring should be done? I am acquiring a Freiden Printer/Punch also. Would this be a better choice? The Freiden is called a "FLEXOWRITER", maybe some information on that? I thank you very much.

Bob Johnson
13320 Winfield St.
Panorama City, CA 91402

New HUG Club Forming

Dear HUG,

Please mention my name in the next issue of REMark, as I am trying to get in touch with other HUGgies forming a Danish/Scandinavian User Group. It could be fun as well to get in contact with U.S. HUGgies, - diskette mail, maybe dial-up connection?

K. Munk Kristensen
2730 Herlev
DENMARK
Phone (02) 91 70 90

Dear HUG,

Concerning the HUG SY: driver mods for NAJAY 2+4 module, there is an error in the reference regarding modification to DKH171.ASM. Specifically, the change at location PBOOT should be deleted, i.e., no change should be made at this location. The modification as originally shown will render disks INITed unBOOTable.

I have noted no other problems. Sorry.

Bruce Terrell
P. O. Box 1922
Pittsfield, MA 01202

Error in HDOS Fig-Forth

Dear HUG,

There is an error in the -CMOVE command supplied with HDOS Fig- Forth:

MOVE on line 3 should be MOV,
0= on line 5 should be 0<,
and the two INXs on line 6 should be DCXs.
Also, <CMOVE is the accepted name for -CMOVE. These changes are in the Forth 8080 Assembler Example No. 2.

The book "Starting Forth" by Leo Brodie of Forth Inc. has two advantages over the recommended "Using Forth" by Forth Inc. First, it is ten dollars cheaper. Secondly, it has been written for beginners, while "Using Forth" has been written like a tech manual.

Michael Mackenzie
Purdue University
P.O. Box 592
Cary Quadrangle
West Lafayette, IN 47906

SIGN UP NOW FOR THE SECOND NATIONAL HUG CONFERENCE

registration on page 9 of this issue

More Info On the Imaginator Graphics Board

Dear HUG,

The review of the Imaginator graphics board in the March Issue of REMark, Imagination Gone Wild by Jim Buszkiewicz, was greatly appreciated. We have been repeatedly told by REMark readers that the review clearly answered any questions they may have had about the graphics board.

Only one comment in the article has caused any confusion. The article stated that the mathematical function pictured on the front cover took 1-1/2 hours to complete. We would like to point out that this was the time required by the host computer in calculating the coordinate data, not the time required by the graphics board to draw the figure. The graphics board is capable of drawing the figure much faster. In fact, by precalculating the data and sending it to the graphics terminal as a condensed data stream, we have drawn the figure in 7.5 seconds.

It is also possible to reduce the host processing time by streamlining the algorithms, a 10 fold increase in throughput is not difficult to obtain.

Peter O. Botten
Cleveland Codonics, Inc.
P. O. Box 45259
Cleveland, Ohio 44145

A "Thanks" For Assembly Language Articles

Dear HUG,

I wish to express my thanks for the article by Pat Swayne in the April Issue of REMark on "Getting Started With Assembly Language". About a year ago, I tried to learn this language and met with all kinds of obstacles. I purchased a number of books, including Heath's ASM Language Programming (EC-1108), Doc Campbell's book, and others, and although I was helped to a small extent, I was hoping for something that could relate to the language most all novices begin with, i.e. BASIC.

I had given up on this until your article appeared and my interest became whetted again. This approach to teaching ASM will bring many HUGgies back who have been frustrated by the other methods.

I sincerely hope you continue this method and hope future articles will be demonstrated using different instructions in BASIC to show how to use assembly language to the fullest.

Alfred M. Fanelli
6504F S. Charter Rd.
Glen Burnie, MD 21061

Library of Public Domain Software Coming to 2nd Nat. HUG Conf.

Dear HUG,

Please let the readers of HUG REMark know that I am planning to bring the library of public domain software to HUGCON '83. Libraries will include SIG/M, CPMUG, and PC/Blue. The SIG/M and CP/MUG libraries are public domain libraries oriented to CP/M users. There are currently 114 volumes in the SIG/M library and 90 volumes in the CPMUG library. Note: volume 55-77 of CPMUG are duplicates of the SIG/M library. I will be bringing the SIG/M and CPMUG libraries on 8 inch IBM standard format, 80 track, double-sided, hard-sectored LLL BIOS-80 format and Heath 40 track, double-sided, soft-sectored format. I will be able to make 40 track single-sided hard- or soft-sectored disks, but that will take longer.

The PC/Blue library currently consists of 28 volumes (I just submitted eleven more volumes that are not in yet) and includes software designed to run on IBM PC and PC compatible (Z-100 ZDOS) machines. The PC/Blue library is on 40 track single-sided, double-density MS/DOS (PC/DOS, ZDOS) diskettes (soft-sectored).

I will be happy to make copies for anyone on their **PREFORMATTED** diskettes at a nominal copying donation. (Last year it was two dollars per volume or a dollar a disk for 3 or more disks per volume.) I will be happy to do the same this year.

Clubs that want to get a large number of volumes of the library should consider bringing a machine for copying and we might be able to work out the same deal we did last year for copying at night. (Last year we went to 4 a.m. Saturday night, I believe.)

I will also be bringing several cases of the New York Amateur Computer Club Catalog of Public Domain Software for those who want them. I heartily recommend that every club get a reference set for all club members to use. Last year there were three volumes. This year volumes four through seven were published, as well as the first volume of the PC/Blue series. These will cost about ten dollars a volume.

I will have these at a booth, hopefully adjacent to the Extended Technology Systems booth, wherever that may be.

Robert H. Todd, Jr.
1121 Briarwood
Bensalem, PA 19020

A Tip For New BASIC Programmers

Dear HUG,

Here is a problem that new BASIC programmers like myself might encounter.

After writing a simple program to test some of the instructions I had just learned, the RUN produced the error response:

AN ILLEGAL CHARACTER WAS ENCOUNTERED

The program was a simple READ DATA, and everything seemed in order according to the book.

It took quite a while of head scratching to discover that, although the program looked right on the screen, internally there was an illegal character in the DATA statement. I had hit the ESC key while entering the number 1. I couldn't see it, but it was there just the same.

Dick Harlow
6 Hayes Rd. #26
Roslindale, MA

More On 4MHz Mods to the H/Z-89

Dear HUG,

I've been reading REMark magazine for some time now and would like to compliment you on your fine work. I've also noticed an overall improvement in recent months and I'm very pleased. Letters and articles from the readers have been pretty interesting lately, and I would like to comment on a couple of them if I may.

First, I would like to say that I'm in full support of new modifications or enhancements that improve the operation of the H/Z-89 (and others) computers, however, from an engineering standpoint, I do not approve of those which (1) take an extensive amount of work or rework, (2) take a departure from the original design that prohibits its use with OEM software/hardware (i.e., a hardwired 4MHz modification that you had to cut foils and add jumpers to). Let me further explain...

Pat Swayne recently put out an article for a 4MHz modification for the H/Z-89 that I had mixed emotions about. I must admit that the circuit design was of good quality and he also provided the necessary patches for modifying your software to support it. My objection to his article concerns the construction and installation of this particular mod. I feel that with just a little more forethought, one could have figured an easier way to do the installation, such as a plug-in modification. Well, one such person did and is marketing it under the name of NAJAY System's 2+4MHz. For $69.95, they provide you with a premanufactured board, Z-80A, one other IC, complete documentation and instructions for installation, and software support for both HDOS and CP/M. It is a plug-in mod that works in both the older 89's and the 89A's. It comes with a 2 year warranty and the one thing that I particularly like is "software/hardware portability" or compatibility. In other words, the modified machine can still use unmodified software (as with Pat's) and the modified software can still be used on an unmodified machine. I guess my real complaint with the hardwired version is the fact that it is so much of a pain in the posterior to restore the CPU board to original condition for troubleshooting purposes or for resale.

In another article, this one written by Bob Small, in REMark Issue 37, concerning the cooling problem with the 89's... It is another prime example of the lack of forethought. Do yourself a favor, save time and don't hack up your cabinet. Just remove your fan, install a filter (Heath P/N 266-1010) and remount the fan so that it is drawing cool air into the cabinet (blowing onto the power switch). For those who have the older machines, you'll also require the newer fan mounting bracket (Heath P/N 204-2452). And if you don't already have them installed, the heat sinks are Heath P/N 215-637 (3 is all you need). As long as I'm on the subject of power supplies, you may also want to prevent a possible "burn-up" by removing the two yellow wires (9 VAC) from P-101, the two orange wires from P-103 to BR-1 (bridge rectifier) and solder the two yellow wires directly to BR-1. This precludes the possible burn-up of connectors and foils of the power supply board.

There are perhaps a few more mods that I could talk about but, I make my living by repairing these machines and I have to keep some integrity over "trade secrets".

Thanks again for a wonderful magazine and the great support that only a HUG could provide.

Joseph B. Travis

Correction to CheapCalc

Dear Hug,

CheapCalc by B. L. McFarland (REMark, February 1983) is a valuable contribution to the literature of public domain computer programs.

Readers who have keyed the program into their computers will find that some corrections will be required to make it useable. Following are the changes I made in order to get the program to run.

```
 135 CR = 0
 230 S$ = ""
 280 FOR I% TO 20: S$ = S$ + CHR$(32): NEXT I%
1230 IF X% >= XM% THEN XM% = XM%
2000 T = T + CW(A): A = A +1:
     IF A <= XM% THEN 1950
2450 SX% = SX% - 1: IF SX% = 0 THEN X% = 1:
     SX% = 1: GOTO 2470
2710 A$(Y%,X%) = "&&&&(" + A$ + =-= + B$ +")"
2890 Y2% = YH: X2% = XH
3780 PRINT BP$: IF CR < 1 THEN XH = X%:
     CR = CR +1: GOTO 2130
3850 N% = VAL(MID$(A$(YH,XH), I = 1):
     IF N% < 1 THEN 3880
```

The program would be considerably enhanced and more easily readable if the techniques of structured programming were employed.

James R. Leverett Jr.
39 Westwood Dr.
Tonawanda, NY 14150

Update On Patches For Z-DOS

Dear HUG,

In the June Issue of REMark, Software Consultation published a series of patches for the Z-DOS utilities FORMAT, MAKE, DSKCOPY, and BACKUP. At the time, the patches were valid for all versions of these programs, but that is no longer true. The patches apply **only** to the versions of these programs with dates **before** 5/27/83 as listed in their directory entry.

Skip Gwyer
Software Consultation Group

Microcomputing and Heath Announce Robot Programming Contest

Dear REMark,

Microcomputing magazine, in association with the Heath Company of Benton Harbor, Michigan, manufacturer of the HERO 1 robot, has announced the first ever Microcomputing/Heath HERO 1 Programming Contest. The HERO 1 is the first personally affordable, educational robot.

The winner in each of two categories (standard HERO 1 and modified HERO 1) will receive a $500 gift certificate which they will redeem for merchandise from the latest Heath Company catalog. There will also be second and third prizes in each category. Entries will be judged on originality and technical feasibility.

Microcomputing must receive all entries by September 1, 1983. Winners will be announced in the December '83 issue. Complete details will be published in each monthly edition of Microcomputing prior to the entry deadline.

Microcomputing is a publication of Wayne Green Inc., headquartered in Peterborough, New Hampshire. Other Wayne Green Inc. publications include 80 Micro, Desktop Computing, inCider, Hot CoCo, 73 Amateur Radio Journal, books for micro users, and Instant Software (applications, simulations, and games programs).

Contact: James Leonard
Peterborough, NH 03458
(603) 924-9471

Correction to the Article "Base Conversion Routine"

Dear HUG,

In reading my latest issue of REMark, the article written by Louis Berger, "Base Conversion Routine", spoke of an unexplainable error. The error is explainable, and changing a few statements will make a very useful program.

The limitation of decimal numbers up to 511 can be increased to 1* 10E6 by rounding. The error is generated by the floating point computations in BASIC. Six digits of accuracy works out to 5.5 digits. To correct for this problem, the statement N=INT(100*(N+.5)/100) will yield a more correct result.

The line numbers to change are as follows:

```
65060 - was D=VAL(B$)
        is  D=VAL(B$)+.1

65130 - was IF D<512 THEN B=2:B$="Binary":GOTO 65140
        is  B=2:B$="Binary"

65135   DELETE

65170 - was D=D-(Y*B^X):NEXT X:PRINT B$;" equivalent:
            ";T$:D=D1:RETURN
```

```
is  D=INT(100*(D-(Y*B^X)+.5/100):NEXT X:
    PRINT B$;" equivalent: ";T$:D=D1:RETURN
```

Richard B. Johnson
235 Pacific Oaks Rd. #205
Goleta, CA 93117

A Patch For Benton Harbor BASIC

Dear HUG,

If you would like to LOAD your programs when running Benton Harbor BASIC rather than having to OLD them, here is a quick fix, and it will only cost you the ability to LOCK. You may still OLD if desired. Old habits are sometimes hard to break.

This patch is for Extended Benton Harbor Basic #110.06.00, and may be accomplished with the patch version of PATCH as presented in REMark magazine.

```
PATCH Issue 50.06.00.

File Name? SY1:BASIC.ABS

Address? 67117
067117 = 230/232
067120 = 114/       Hit CR
067121 = 117/       Hit CR
067122 = 103/101
067123 = 113/104
067124 = 327/^D
Address? ^D
PATCH Issue 50.06.00.

File Name? ^D
>
```

Daniel A. Schlichtig
18832 Cabral St.
Canyon Country, CA 91351

More on Christmas Graphics Program

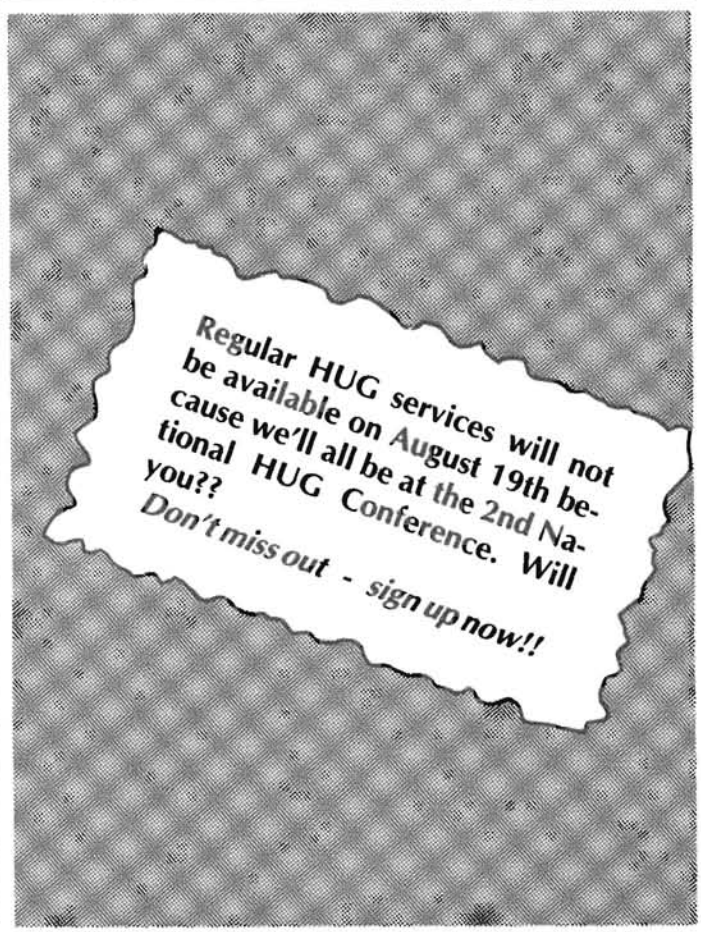
Dear Jennifer McGraw,

I received REMark Issue 38 yesterday and was interested to see your comments about your Christmas Graphics program. I run MBASIC with CP/M, and had the same problems; terminal beeps and messed up screen when the picture was moved on the screen. I added do-nothing for loops to get it running smoothly.

However, I do not believe that the problem was caused by MBASIC confusing the codes, but rather by the input FIFO in the H19 terminal overflowing. The terminal is busy moving the picture around, and can't take characters out of the FIFO as fast as they are being sent over from the computer, so there is an overflow.

When FIFO overflows, the terminal beeps to let the operator know that there is a problem. The overflow caused lost data, hence the messed-up screen. This condition should not occur, because the H19 is supposed to transmit an X OFF shortly before the FIFO buffer sets full, and then an X ON as soon as it can handle more data. Until I read your letter in REMark, I thought the problem was in my equipment or software because I assumed that the program had been checked out by HUG before being published. So it seems that either the terminal is not doing it's job with the X OFF, X ON sequence, or else the system is ignoring them.

I know that the trouble is not due to gremlins in the ICs, because I left lines 10 and 15 in the program, exactly as presented in the article!!



Daniel A. Schlichtig
18832 West Cabral St.
Canyon Country, CA 921351

A Problem For the H8 User

Dear HUG,

I happen to be one of those loyal H8 users, using the H17 disk drive. I am comfortable with the system and it has generally served me well. However, I have been plagued with problems with reliability of data processing. In using MBASIC programs, I frequently get Error #54, 55, 57, or 65. The system opens and closes files at will, and at rare times, will dismount the disks. CP/M simply tells me that I have a Disk I/O error of a particular type.

The problems are intermittent, but occur at the worst possible times. After much search and analysis, I have finally traced the Disk I/O problems to the programming plug on the circuit board of the disk drive. The IC socket used to receive the programming plug is not reliable in my view, so I carefully removed it, and soldered jumpers at the appropriate points. I have had no I/O problems since.

Bob McIntosh
1931 Geneva St.
P. O. Box 1113
Souix City, IA 51102

Getting CP/M Console Swap Program to Work With the H/Z-100

Dear HUG,

Its been pointed out that my CP/M console swap program (REMark #33) won't work on the H/Z-100 as written if you are using a serial printer. The TTY: port is used by our BIOS to drive a serial printer and thus would be unavailable for use by SWAP. Well, a user (whose name I don't know) pointed out a way around this. It seems that the guys who wrote the CP/M-85 BIOS found a useful designation for the last I/O BYTE option for the console (UC1:). They programmed it to drive the modem port (serial B) as an optional console and so we can use it as the alternate device of TTY:. To do so requires a one byte change in my program and the use of CONFIGUR in CP/M-85. The changes are:

1) In the program (in the IOSWAP), change XRI 01H to XRI 02H, and then reassemble and load the program.

2) In CONFIGUR, select the modem option and then simply set up the baud rate and other parameters to meet your needs. Please note that on the connector (serial B), you must have pin 6 jumpered to pin 5 to pull this signal up.

I hope this information is helpful.

Marc O. Aagenas
Software Consultation Group

A Program to WRITE Out a Message

Dear HUG,

I have included here a program which simply writes out a message on the screen, not prints mind you but WRITES out the message. This will show how the PRINT @ statement used in the TRS-80 computer is used. Also shown in the program is the conversion functions required to bring this about on the H/Z-89 computer. The TRS-80 screen is divided into 1920 bytes (not including the 25th line) of information. I use cursor positioning to do the same as PRINT @ in TRS-80 language (i.e., PRINT @ 513;"ABC" might be typed PRINT FN B$(641);"ABC" to get the same result).

The following formulas will give a direct conversion of PRINT @:

```
Def FN A$(A1,A2)=CHR$(27)+"Y"
              +CHR$(31+A1)+CHR$(31+A2)
Def FN D1(A3= INT ((A3/64) * 1.5)
Def FN D(A4)= INT ((A4-(INT(A4/64)*64))
                                  +1*1.25)
Def FN B$(A5)= FN A$(FN D1(A5),
                                  FN D(A5))
```

Using these formulas, you may then type PRINT FN B$(513);"ABC" if you see PRINT @ 513;"ABC" in a TRS-80 program, instead of having to find where the information is printed on the TRS-80 screen and shrinking your screen to their dimensions. I hope someone will find this program useful.

Kenneth B. Blois
PSC # 1 Box 2039
APO SF 96366

FREEZE Command for Benton Harbor BASIC

Dear HUG,

The improved FREEZE command for Benton Harbor BASIC in Pat Swayne's article "Improvements to Benton Harbor BASIC", REMark Issue 29, June 1982, and HUG Disk #885-1119 are just super. It saves considerable time in loading large programs and data files. I use a Benton Harbor BASIC mail label program that takes eight minutes to load 200 names and addresses into memory from a data file using delimiters and the MATCH command. Eliminating the delimiters and MATCH calculations and using the INPUT command reduced the time to load 200 names to about six minutes. However, using the FREEZE and UNFREEZE commands reduces the time required to load the program and data to just eight seconds. Certainly an improvement!

But there is a small penalty to pay. The program file uses 39 sectors on disk. The name and address data file uses 47 sectors for a total of 86 sectors. The file containing both program and data created by the FREEZE command uses 104 sectors. I believe this is a small price to pay for the eight second load time. Now if Pat would consider adding the PUT and GET commands found in cassette BASIC, we would have the same speed and probably take no more disk space than the program and data files together.

Robert A. Naegele
15w780 Fillmore
Elmhurst, IL 60126

```
00005 E$=CHR$(27):E1$=E$+"E"
00010 DEF FN A$(A1,A2)=E$+"Y"+CHR$(31+A1)+CHR$(31+A2)
00015 DEF FN D1(A3)=INT(A3/80)
00020 DEF FN D(A4)=INT(A4-(INT(A4/80)*80))+1
00025 DEF FN B$(A5)=FN A$(FN D1(A5),FN D(A5))
00030 PRINT E1$;E$;"x5"
00035 READ A
00040 IF A=-99 THEN 55
00045 PRINT FN B$(A);"%"
00050 GOTO 35
00055 PRINT FN B$(1690)
00060 PRINT E$;"y5":END
00065 DATA 83,163,243,323,403,244,245,326,406,407,408,409
00070 DATA 330,251,171,170,169,249,411,412,413,334,255,175,95
00075 DATA 94,174,254,335,416,417,338,259,179,99,98,178,258,419
00080 DATA 420,421,342,263,184,185,343,424,425,346,266
00085 DATA 192,113,114,194,274,354,434,275,196,117,356,437
00090 DATA 438,439,440,361,281,200,279,360,441,442,443,364,285
00095 DATA 366,446,287,288,369,449,450,371,292,293,374,454
00100 DATA 295,296,377,457,458,459,460,381,301,220,299,380,461
00105 DATA 462,463,464,385,306,226,146,387,468,469,470,471
00110 DATA 391,311,231,151,312,313,393,474,224,225,227,228
00115 DATA -99
00120 REM Writes out "hello Kenneth"
```

Dear HUG,

I have always been envious of the APPLE programmers whose programs appear in other magazines. They easily call out their disk catalog to see what filename they want or may be in use.

The problem with Extended Benton Harbor BASIC is:

1. If, in running a program that you intend to manipulate several data files, you forget your input filenames, you are in trouble. You start guessing and get error messages or you exit BASIC and call CAT.

2. If you are out-putting a file that you name interactively and forget names already in use, trouble again. Either exit BASIC losing everything or risk losing data in a write error to a filename already in use.

I don't pretend to understand HDOS or all the information in DIRECT.SYS which it puts on each disk. However, it does have several regular features that are accessible and useable in BASIC:

1. Information is stored as ASCII decimal code.
2. Filenames start the file occupying 11 character positions.
3. Filenames alternate with 12 character blocks I don't understand, yet.
4. Filenames are right justified with extension, .ext, left justified. Positions in between are filled with nulls (DEC 0).
5. Blank filename fields are marked with a DEC 255 and are otherwise filled with nulls.
6. All the optional filenames (which excludes RGT, GRT, and DIRECT.SYS itself) are always at the high end of DIRECT.SYS.

The simple subroutine below will read the directory and print on the console all files with extension '.DAT', the EBH BASIC default extension for BASIC data files. Obviously, the program can be used to search for anything in the directory you want to look for. The subroutine terminates when it comes to a blank filename field.

Thought this programming hint might be of interest to REMark readers.

Joseph R. Bobbitt III
238 Christopher Street
Upper Montclair, NJ 07043

```
12994 REM
12995 REM    Subroutine for reading BASIC data file titles.
12996 REM
12997 REM    Open the disk directory for reading.
13000 OPEN "DIRECT.SYS" FOR READ AS FILE #1
13005 REM    Clear out string.
13010 A$=""
13015 REM    Read first file name block.
13020 FOR I=1 TO 11:A=CIN(1):A$=A$+CHR$(A):NEXT I
13025 REM    Is filename block empty?
13030 IF LEN(A$)<=2 THEN CLOSE #1:RETURN
13035 REM    If the file name has extension 'DAT', print it.
13040 IF RIGHT$(A$,3)="DAT" THEN PRINT LEFT$(A$,LEN(A$)-3)
13045 REM    Skip to next file name block.
13050 FOR I=1 TO 12:A=CIN(1):NEXT
13055 REM    Go around again.
13060 GOTO 13010
```


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

Saint Joseph, Michigan 49085

BULK RATE
U.S. Postage
PAID
Heath Users' Group

Volume 4, Issue 7

POSTMASTER: If undeliverable, please do not return.

885-2042