REMark

Issue 26 • March 1982

Official magazine for users of Heath computer equipment.

on the cover . . . .

THE NORTH PIER — St. Joseph, Michigan

By: Kent Wallace

on the stack

>CAT

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

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

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

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

Send payment to:

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

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

REMark

. . . AND THEN I WROTE . . .

SEE YOU IN CHICAGO AND . . .

Hopefully, by the time the ink drys on these words, Lake Michigan will not look like it does in the cover shot this month. As February begins, it looks like a repeat of January that robbed us of 303 degrees of normal temperatures and dumped 32 inches of partly cloudy on us.

There are, however, rays of sunshine outside and some pretty exciting planning going on inside.

REGIONAL HUG MEETINGS

We plan to visit several cities this year and have REGIONAL HUG meetings. This will provide an opportunity for neighboring clubs to get acquainted and get up-dated on all the latest rumors from Benton Harbor. When possible and timely, new hardware and software will be unveiled at these affairs and, of course, plenty of time for questions and answers, plus door prizes. Our first meeting will be in San Francisco at the West Coast Computer Faire Sunday, March 21. Check the Heath or HUG booth for room number in the Civic Center. Currently on the agenda are the cities of Anaheim, Denver, St. Paul, St. Louis, Omaha, Cleveland, Pittsburg, Houston, and Alexandria Va. The schedule is being firmed up and each club will be notified by mail or in these pages. The members of the above named host clubs are asked to contact neighboring clubs to determine probable attendance and begin thinking about location and desired agenda items not already mentioned.

NATIONAL HUG CONFERENCE

What began as a suggestion and perhaps only wishful thinking one night on the HUG bulletin board, is in fact going to happen. The first annual NATIONAL HUG CONFERENCE will be held this summer in Chicago on August 6, 7, and 8 at the Hyatt Regency O'Hare Hotel.

Current thinking is that we will begin with registration and an informal "glue the faces on the names" session on Friday evening. Saturday will be a full day of meetings, demonstrations, exhibits, lectures, questions and answers, dinner and door prizes. Many of the Heath and Zenith executives will be on hand to chat as well as engineer types for brain picking. Sunday morning will be free for looking at exhibits, playing with new classified toys or whatever. Sunday afternoon will be more of Saturday with activities ending in the early evening for those who need to catch a plane home.

==========

"... exhibits, demonstrations,

Q&A, dinner and door prizes."

==========

Tentatively, we may bus over to Michigan for a tour of HEATH on Monday, if there is sufficient interest. What we need as soon as possible is a head count of those who plan to attend. Your cost is transportation, lodging, $15 registration and some meals. We're trying to get someone we all know (and love) to take care of Saturday and perhaps Sunday meals. Hotel reservation information will be detailed in next months magazine and we hope will be as painless as possible. A "how much are you interested card" is on the last page for you to fill out (or copy) and send in to us. Vendors are urged to show their wares at no cost, but we need to know how many. (Eight foot tables will be supplied.) How about you vendors popping for coffee and rolls Saturday morning? Just thought I'd ask!

We will be in contact with a few club officers to ask for assistance with all the details.

ADVERTISING

In the unlikely event you overlooked the fact that all the stuff mentioned here is going to cost a bag of money, I need to point to the increase in postal rates as only one reason we have decided to add eight pages to the center of **REM**ark, reverse a long standing decision and accept display advertising beginning with this issue. All ads must be camera ready and accompanied by full payment. Frequency discounts are earned as you go along. Rates and other information areavailable upon request or over the phone. Deadline for each issue is 60 days preceding the desired month. **REM**ark is mailed the first week of each month.

$ $ $ SOFTWARE $ $ $

Over the weekend, I got to read the January issue of **CHUG** (Fairfax Va.) wherein President Larry Henderson reported on his recent visit to another HUG group on the west coast. It seems that one of the highlights of the meetings is to copy HUG software and then have the guts to charge

each member two bucks for each disk. Larry rightfully pointed out to club officials that the sale of HUG software is the primary source of income which keeps HUG afloat and more software going out the door. The 18 buck membership dues puts 12 issues of REMark in your mailbox, prints and mails one software catalog and maybe swallows one HUG salary. Most of the other expenses are covered by the sale of software at an average of 5 or 6 bucks a program. And that is with Heath handling the xeroxing and packaging of the disks, all the order entry and shipping without charge to us! The fact that most, if not all of the software, is copyrighted and that copying copyrighted material without the permission of the rightful owner is prohibited by law is another matter.

==========

"... copying copyrighted material without the permission of the rightful owner is prohibited by law."

==========

Granted, most of the HUG authors could care less if you copy their program and give it away, but he contributed it to HUG to help HUG and the other members. (There will be a short pause, while I get down off the soap box.)

Last month we introduced a plan to pay royalties on the sale of selected programs written by you and distributed by HUG. The response has been good. However, don't get bent outta shape if we turn you down. It must be financially rewarding for both of us.

One of four things will happen to a program sent to us.

1. The program is short and useful or demonstrates unusual programming techniques that will interest readers of **REM**ark. Thank You!

2. We will return all your materials if we don't plan to use the program in any fashion in the near future. (How many Hangman games can you deal with?)

3. We THINK we will group the program with others as a regular HUG release in the near future. In that event, we will send you a complimentary copy of the latest HUG software release, (mainly to replace the disk you send us) and renew you for another year.

4. We KNOW we will use the program and are willing to pay royalties on it and will send you a copy of the software license agreement for your signature.

The main thing is, we want to acknowledge receipt of your program within a couple days and reduce the number of filing cabinets holding down this end of the building. So keep those cards and letters coming!

:JB:

SPEED ON HEATH MICROS

by Alfred D French
201 Central Avenue
Jefferson, LA 70121

There has been a lot in BYTE and other computer magazines on the speeds of different languages and different machines. In many cases, raw speed is less important than other factors, such as easy program development. Many times, however, high speed is crucial to the success of the program.

Over the past two years, I tested different languages and different computers. It was good exercise because I learned several languages. On the basis of these studies, I recently bought a second computer system for the molecular modeling studies that I do in my work as a research chemist.

The benchmark program was by my brother, Walt French. As usual, the program finds the prime numbers between 1 and 10,000. The prime numbers are determined by division, unlike Jim Gilbreath's method in the September 1981 BYTE, page 180. Because of the use of multiplication, division and square roots in my tests, ranking for the languages was different than Jim's.

As our method used square roots, only languages with that function were tested. Since scientific programs depend on such capabilities, it was a fair test.

The program in Benton Harbor BASIC is:

```
10 FOR I=3TO 9997STEP 2
20 I2=SQR(I)
30 FOR J=3TO I2 STEP 2
50 IF INT(I/J)*J=I GOTO 70
60 NEXT J
61 PRINT I
65 K=K+1
70 NEXT I
75 PRINT K
80 END
```

The programs in other languages were as close to this algorithm as possible.

Of the computers that I tried, the fastest time was 0.5 seconds on an IBM 3033 mainframe, using a PL-1 compiler. The slowest was a TI-59 programable calculator, which needed 17 and one half hours. These were the results on an H-89, running a standard 2 mHz clock:

Language	Time (Min:Sec)	Language Type
CBASIC	35:57	P-Code Compiler
BHBASIC	31:21	Interpreter
MBASIC	23:30	Interpreter
MBASIC*	17:57	Interpreter
MBASIC	9:20	Compiler
FORTRAN*	7:05	Compiler
UCSD PASCAL*	6:50	P-Code Compiler
MBASIC*	4:14	Compiler
PASCAL 3.0*	3:05	Compiler

*In these tests, integer variables were used where possible.

The P-Code Compiler and run-time interpreter of UCSD Pascal were very efficient, compared to FORTRAN, for instance, which is native code compiler designed specifically for number-crunching problems. In Jim Gilbreath's test, UCSD Pascal took nearly 15 times as long as FORTRAN. Pascal 3.0 is an early version of Pascal MT+, a very powerful native code compiler now available under both CP/M and HDOS. Although the Benton Harbor program was easiest to write, the total time to write, compile and execute the Pascal MT program was less than the comparable BHBASIC time.

I won't mention all the non-Heath computers, but the DEC mainframes were in the :03 and :10 range using FORTRAN and Pascal. A Hewlett-Packard HP1000 mini took 2:45 and a PDP 11/03 needed 10:41 for their interpreted BASICs. If one has to buy time on the big machines at commercial rates, Heathkits are clearly cost-effective.

If more speed is needed, an obvious tactic is to change languages or to compile an interpreted language such as with the MBASIC compiler. There are other methods, however. One might use a language with less power if you don't need floating point arithmetic and so forth. Two good examples are the Tiny Pascal compiler from HUG and the C/80 compiler from the Software Toolworks. The Andex Industries BASIC compiler is another such program. On some text-handling programs, Tiny Pascal was far faster than the BASIC interpreters. It was also as fast as or faster than the big compilers and much less expensive.

A less obvious way to get more speed is in the algorithm. Clearly, it helped to use integer variables where appropriate. Another helpful change not included in the above table was to use the MOD function instead of the division and multiplication in line 50. That sped up the program by one third in interpreted MBASIC. Even the output format is critical. In the first FORTRAN program, the list of output numbers was spaced over 5 spaces. This added nearly 30 seconds to the final time given above.

My big modeling program had to stay in FORTRAN. That preserves maximum portability to other scientist's computers and avoids translation of a lot of code. Since I was running series of overnight runs, more speed was clearly worthwhile. Therefore, I bought an H-8 computer and replaced the 8080A cpu with a DG Z80 4 mHz cpu and installed the DG 64K board. The doubled clock speed made everything go twice as fast. I made no attempt to take advantage of the extra speed of the Z80 over the 8080A due to the extra instructions available to the Z80. The Microsoft FORTRAN compiler does not support that. The other change for more speed was to purchase the HA-8-3 Color Graphics board.

The reason for buying the Color Graphics Board is that it has a socket for an arithmetic processor chip. There are other boards for the H-8 which use the same chip but the additional features of the color board made the extra price worthwhile. The reason the color board is not supplied with the chip is that the chip alone is expensive. We found one that would run at 4 mHz for $150.

The arithmetic chip does its work with hardware instead of the software libraries of FORTRAN and Pascal. In order to use these chips, one needs new subroutines that can replace the standard arithmetic functions. The routines are available for HDOS FORTRAN (from Microtran, 76 Flintwell Way, San Jose, CA 95138) and come with Lucidata Pascal and MT+ Pascal. The port assignments in the software might not match the port assigned on the board for the arithmetic chip. Some changes, either in hardware or software, might be needed. Heath's standard port is 274.

How much faster are programs using the arithmetic chip? On the H-8 described above using FORTRAN, 1:46 was needed to find all the prime numbers. That is a two-fold improvement due to the chip and two-fold for the 4 mHz. This same

Vectored to page 13

PASCAL CORNER — PART IV

by Henry E. Fale
QUIKDATA COMPUTER SERVICES, INC.
2918 S. 7th. St.
Sheboygan, WI 53081
(414) 452-4172

Welcome to part four of the Pascal Corner. I want to start right off by printing a brief comparison of Pascal Compliers available to Heath Users as submitted to me by Fred Pospeschil. This was promised last month, so here it is. I feel it may help you in knowing what is available, and some features of each, so you can make an intelligent decision on what version will be most suited to you.

PASCAL COMPILERS FOR HEATH COMPUTERS

Like Henry, I consider myself to be a beginner, or novice, Pascal programmer. However, over the past several years I have had the opportunity to use Tiny, UCSD, Lucidata, and MT MicroSYSTEMS MT+ Pascals. The purpose of the following remarks is to provide a brief comparison of, in very general terms, these implementations of the Pascal language and to respond to frequent questions posed by fellow Heath users around the country.

Pascal implementations fall into two general categories - Pcode and native code systems. The Pcode compilers take the Pascal code and compile it into an internal "Pascal Machine" code - thus the term "Pcode". This Pcode is then interpreted by a run-time package in a manner similar to CBASIC. UCSD and Lucidata Pascal are of this type.

Native code implementations compile and translate the Pascal source code all the way to a binary load module, an .ABS file, which will execute without any separate runtime interpreter. In this case, any necessary run-time I/O or math functions are included in the .ABS file. Tiny and MT+ Pascals are members of this group.

As one might expect, the native code compilers produce faster running load modules than do the Pcode implementations. Pcode programs, however, usually run three to four times faster than fully interpreted languages such as BASIC. Actual speed comparisons are dangerous at best, however, questions along this line seem to be the ones asked most often.

Thus, based on running several different CPU intensive programs, the following numbers appear to be reasonable comparative figures. If the BASIC version takes 180 seconds, then Lucidata uses 45, UCSD uses 30, Tiny Pascal uses 8-10, C80 about 6-8, and FORTRAN and Pascal MT+ about 2 seconds.

When other than simple programs are attempted these numbers go out the window. For example, if you are working with a graphics application on the HA-8-3 then the Lucidata graphics compiler is almost as fast as Tiny Pascal and if the application uses line drawing then it is actually faster. Since most graphics programs require trig functions, Tiny Pascal has only limited usefulness in this area. If computational speed is important then Lucidata and MT+ are the preferred choice as they both support the AMD9511 math chip.

When considering computational work the compilers vary considerably. Tiny Pascal is integer only - in fact integer is the only data type supported. UCSD has both single and double precision floating point, Lucidata has nine digit software and 6.5 digit hardware (graphics version only) floating point support. MT+ has 6.5 digit software and hardware capabilities for scientific work and 18 digit BCD reals for business applications. Isn't it nice to have a lot of choices?

The four compilers also require different amounts of disk space. Tiny Pascal being "Tiny" runs well on a single drive system - especially if run STAND-ALONE. Although source code and the .ABS output can be on other drives, Tiny Pascal assumes that the compiler/translator are on SY0:. UCSD is considerably larger and needs at least two drives. Unlike the other compilers, UCSD is a complete system. It provides its own text editors, assembler, linker, utilities, and an operating system which is incompatable with HDOS, CP/M, the ADM9511, and 400K drives (as of this time). Lucidata runs well on a one drive system and like Tiny and MT+ is compatable with HDOS, 400K drives, and all Heath compatible text editors. Pascal MT+ is a big system. Although it will run on a two drive system (100K drives) three are recommended and two 100K and a 400K drive make a very nice operating environment. As is usually the

case, more disk space makes things nicer.

Another area in which there usually seems to be some interest is load module size. This area may seem confusing due to comparing Pcode files produced by Lucidata and UCSD with .ABS files created by Tiny and MT+. In general, the Pcode files will tend to be smaller than the .ABS files for the same program. However, the Pcode file must have a run-time package in memory before it can be executed. For one small program I found that Tiny Pascal's .ABS was 12 sectors, MT+'s .ABS was 15 sectors and Lucidata's .BIN Pcode file was 3 sectors. The standard Lucidata run-time package is about 50 sectors. This includes floating point math support which Tiny Pascal never supports. Since the program only used integers I did not link in the floating point routines when using MT+. As I remember, UCSD Pcode files were in the ball park with Lucidata sizes. These comparisons are only general as UCSD uses a different way of reporting disk usage than does HDOS and CP/M. Since there are some considerable differences in the capabilities and manner of implementation between these Pascals, solid comparisons of load module size are for the most part, not easily made or practical.

The four compilers also vary considerably with respect to speed of compilation. Tiny Pascal is without a doubt the fastest due to being written totally in ASM and it only has to worry about a small subset of the language. UCSD and Lucidata are about the same speed and much slower than Tiny Pascal. This should be expected as they are Pcode systems and both implement most of the language, with Lucidata handling somewhat less than UCSD. Pascal MT+ may be a little slower, in total, however it implements the full ISO standard, including conformant arrays, as well as many other powerful extensions. With MT+, comiplation time includes generation and linking of MicroSoft compatable relocatable load modules. Although this makes the total compilation phase take a little longer the load modules run faster. If you use the modular compilation feature of MT+ the total time may be the same or even a little faster. As usual, you have to pay for speed. It's mostly a question of when you pay -- during compilation or during program execution.

What about changes and system upgrades? As far as I know, Tiny Pascal is not projected for any major upgrades thru HUG, however, members of our local club have been working on several useful extensions. These may be made available in the not to distant future. (If they are, I'll keep you informed -- Henry). The word I've heard is that UCSD 4.0 will not be available for 100K drive systems. Just what it will require has not been released. Lucidata Pascal is undergoing a major upgrade to version 3.0. This will add pointers, trig functions and more ISO standards compatibility amoung other things. The graphics version just recently announced is a major upgrade for those using the HA-8-3 graphics board. The HDOS version of MT+ is being upgraded before its initial release. In addition to several extensions to facilitate interfacing with HDOS internals the initial release should also have 32 bit integers and a library manager for your relocatable load modules.

So there you have it -- a general overview which does considerable injustice to four software packages all of which can be useful and cost effective if used properly and within their design limits. I can't remember how many times I've been asked "Which one is best?". Actually none is "best" as they are all tools and which is most cost effective depends on the task at hand. For integer only applications which do not require complex structures/records Tiny Pascal can be very satisfactory. It is a good implementation for initial exposure to the language. As soon as your application requires floating point math, trig, records, etc.; you must move to one of the other implementations. Lucidata has considerably more capabilities at a modest price. UCSD has still more capabilities but the cost is about three times as high (and the support is not as good -- Henry). If you have, or need, larger capacity disk storage then Lucidata is clearly a product you should look at. If you need the full language and want maximum access to your system then MT+ stands tall. From my perspective, MT+ is also the best choice for commercial business applications (18 digit BCD reals) or when maximum system speed is needed or desired. MT+'s major drawback is that it requires somewhat more system resources than do the others and that costs money. I have heard MT+ described as "a high level language for assembly language programmers". After Beta testing it for over nine months I tend to agree.

If you are not totally satisfied with your current language(s) or are just interested in learning Pascal, I hope the above comments and observations, though probably biased, will be of some assistance in your decision making process and help you to avoid buying an implementation which is not suited to your purposes. -- Fred

I thank Fred for those comments and observations. I wanted to include this writeup in this month's corner because I feel knowing the various Pascals, with their

limitations and benefits, is as important as knowing the language. Now moving right along.

NEW CONCEPTS

We will be introducing some new concepts in this section to make Pascal more useful. First off, I want to discuss the output of data to a printer device. I will assume here that a line printer (LP:) is connected to the computer. If you are using an alternate terminal (AT:) or any other, make the appropriate changes.

PRINTER AND DISK I/O

When we can output to a printer and do disk I/O our programs become more useful. The disk I/O will be saved for a later section. To output to a printer in Lucidata Pascal, it is a good idea to load the printer device from HDOS the same as you would from MBASIC. This is not required, but will speed things up. From the HDOS prompt, you would simply LOAD LP: . When running the actual program, you must specify the printer device on the PRUN command line. If you noticed in our last program, TAXRATE, the command line was simply PRUN,TAXRATE. In our new program, LOAN, since we will be using a printer, this must be specified. The command line will look like this PRUN,LOAN,LP:. If this LP: is omitted, there will be no output to the printer, and the program is likely to simply stop when it reaches the WRITE statements in our program.

Now that the printer has been declared in the PRUN command line, it must also be declared in the VARiable section as follows:

```
VAR  PRINTER : FILE OF CHAR;
```

This tells Pascal that the variable named PRINTER is a variable consisting of characters. To output a blank line to the printer is similar to that discussed previously for output to the console, with the exception that the variable PRINTER is an argument of the statement, for example:

```
WRITELN(PRINTER);
```

will output a blank line to the printer device. I should also mention at this point that the printer device must be opened for output before data is sent to it, again, similar to MBASIC. The way we do this, is:

```
REWRITE(PRINTER);     (* open LP: for write  *)
```

Now if we want to write data to the printer, it's about the same. Let's assume we want to output the real variable TOTALINTEREST to the printer, and this variable has been previously declared. We also want to output a title with it. This would be done as follows: WRITELN(PRINTER,'TOTAL INTEREST =$',TOTALINTEREST:9:2);

If the variable TOTALINTEREST contained 1245.323 the output would look like this:

```
TOTAL INTEREST =$   1245.32
```

In past installments of this series we talked about declaring variables and constants at the program start. These variables and constants then apply to the entire program, and are said to be GLOBAL values. Say you have a procedure that uses several variables for a calculation, and they are not used elsewhere. Instead of declaring them GLOBAL at the program start, they can be declared only for that procedure (or function), providing they will not be used outside that procedure (or function). One advantage to this is the same variables can be used for different things in different procedures or functions. Another, is it will keep things clearer if they are only declared where they are used. I will now present in whole my program LOAN and will then discuss points we have not covered thus far. This is a simple program to calculate the total interest on a loan and the payments per year for that loan when the principal borrowed, interest rate, payments per month, and term in years is the input.

```
(*   PROGRAM LANGUAGE--LUCIDATA PASCAL
     PROGRAM TITLE: LOAN PAYMENT

     PROGRAM SUMMARY--CALCULATES REGULAR PAYMENT ON A LOAN.  OUTPUTS DATA
     TO A LP:.  IT IS A GOOD IDEA TO LOAD LP: PRIOR TO USE TO SPEED
     THINGS UP, ALTHOUGH IT IS NOT REQUIRED.                          *)
```

```
PROGRAM LOAN (PRINTER);            (* DECLARE THE OUTPUT DEVICE HERE *)

     (* COMMAND LINE FOR RUN TIME WILL BE ON ONE LINE AS FOLLOWS:
        PRUN,LOAN,LP:                                              *)

CONST     ESC       =27;           (* H19 ESCAPE FUNCTION CODE *)

VAR       PRINCIPAL, ANNUALINTEREST,
          REGULARPAYMENT, TOTALINTEREST      : REAL;
          PAYMENTSPERYEAR, TERMINYEARS       : INTEGER;
          A : CHAR;
          FINISHED : BOOLEAN;

          PRINTER  : FILE OF CHAR;

PROCEDURE CLEARSCREEN;
    BEGIN
        WRITE(CHR(ESC),'E')
    END;   (* CLEARSCREEN *)

PROCEDURE GETDATA;
    BEGIN
       CLEARSCREEN;           (* NOTE ONE PROCEDURE CALLS ANOTHER *)
       WRITELN('                   ** LOAN PAYMENT **');
       WRITELN;
       WRITELN;
       WRITE('ENTER AMOUNT OF LOAN: ');
       READLN(PRINCIPAL);
       WRITE('ENTER ANNUAL INTEREST: ');
       READLN(ANNUALINTEREST);
       WRITE('ENTER PAYMENTS PER YEAR: ');
       READLN(PAYMENTSPERYEAR);
       WRITE('ENTER TERM IN YEARS: ');
       READLN(TERMINYEARS)
    END;  (* GETDATA *)

PROCEDURE CALCULATE;
VAR
     TEMP, INTERESTPERIOD     : REAL;
     NUMBEROFPAYMENTS         : INTEGER;

     PROCEDURE POWER;    (* NOTE A PROCEDURE INSIDE A PROCEDURE *)
     VAR  X    : REAL;
          Y    : INTEGER;

          BEGIN    (* POWER *)
               X := INTERESTPERIOD +1;  (* THIS ROUTINE IS USED TO  *)
               Y := NUMBEROFPAYMENTS;   (* RAISE X TO THE Y POWER   *)
               TEMP := 1.0;             (* FOR X>0 AND STORES ANSWER*)
               FOR I := 1 TO Y DO       (* IN TEMP FOR LATER USE IN *)
               TEMP := TEMP *X;         (* CALCULATING PAYMENT      *)
          END; (* POWER *)

     BEGIN (* CALCULATE *)
          INTERESTPERIOD := (ANNUALINTEREST/100)/PAYMENTSPERYEAR;
          NUMBEROFPAYMENTS := PAYMENTSPERYEAR * TERMINYEARS;

          POWER; (* CALL POWER PROCEDURE *)

          REGULARPAYMENT := PRINCIPAL * INTERESTPERIOD/(1-1/TEMP);
          TOTALINTEREST := REGULARPAYMENT*NUMBEROFPAYMENTS -PRINCIPAL
     END; (* CALCULATE *)

PROCEDURE DONE;
     BEGIN
          WRITE('DONE?  ');
          READ(A);
          IF (A='Y')
               THEN FINISHED := TRUE
               ELSE FINISHED := FALSE;
     END; (* DONE *)
```

```
PROCEDURE PRINTANSWER;          (* OUTPUT TO CONSOLE AND LP: *)
     BEGIN
          WRITELN;
          WRITELN;
          WRITELN('REGULAR PAYMENT =$',REGULARPAYMENT:9:2);
          WRITELN;
          WRITELN('TOTAL INTEREST ON LOAN =$',TOTALINTEREST:9:2);
          WRITELN;
          WRITELN;              (* ALL DONE WITH CONSOLE OUTPUT *)
          REWRITE(PRINTER);     (* OPEN LP: FOR WRITE--NOW FOR PRINTER *)
          WRITELN(PRINTER);
          WRITELN(PRINTER,'AMOUNT OF LOAN =$',PRINCIPAL:9:2);
          WRITELN(PRINTER);
          WRITELN(PRINTER,'ANNUAL INTEREST RATE =',ANNUALINTEREST);
          WRITELN(PRINTER);
          WRITELN(PRINTER,PAYMENTSPERYEAR,' PAYMENTS PER YEAR');
          WRITELN(PRINTER);
          WRITELN(PRINTER,'PRINCIPAL LOANED FOR',TERMINYEARS,' YEARS');
          WRITELN(PRINTER);
          WRITELN(PRINTER);
          WRITELN(PRINTER,'REGULAR PAYMENT =$',REGULARPAYMENT:9:2);
          WRITELN(PRINTER);
          WRITELN(PRINTER,'TOTAL INTEREST ON LOAN =$',TOTALINTEREST:9:2);
          WRITELN(PRINTER);
          WRITELN(PRINTER);
          REWRITE(PRINTER)              (* CLOSE LP: *)
     END; (* PRINTANSWER *)

BEGIN  (* MAIN PROGRAM STARTS HERE--PROGRAM BODY  *)
     FINISHED := FALSE;
     REPEAT
          GETDATA;
          CALCULATE;
          PRINTANSWER;
          WRITELN('THIS IS THE END OF THE PROGRAM');
          DONE;
     UNTIL FINISHED
END.       (* MAIN PROGRAM END *)
```

PROGRAM EXPLANATION

That's it. You can see the programs are getting more complex. I will not cover what we have already covered to save time and space. The first new item we encounter is at the program name. In parenthesis any output device, and that includes disk files, must be listed here. It tells Pascal that the program will be using and declaring an output device. I choose the name printer for this, which also shows up in VARiable as FILE OF CHARacter;. That name is not significant until defined, you could have choosen BITBUCKET if you liked. Again note the next comment line. After the LOAN.PAS is compiled to LOAN.BIN, when run, the LP: must be specified on the command line as shown.

The next procedure, CLEARSCREEN, we have seen and explained before. The only new item in the next procedure GETDATA is that it is a procedure which is calling another procedure. This is perfectly legal, and you will be seeing more and more of it. If you start serious writing you will use it much yourself. This could have been done in the main program body before calling GETDATA, but this is just as good, and perhaps more logical. It's a matter of choice.

Procedure CALCULATE contains some new concepts in that it contains a nested PROCEDURE POWER, and each of these procedures contain LOCAL VARiables. Since these variables are only used within these procedures, they were declared here, and not GLOBALLY at the start. Also note that the VARiables X,Y, and I, are only used within the PROCEDURE POWER, and are thus declared, but the VARiables TEMP, INTERESTPERPERIOD, and NUMBEROFPAYMENTS are used in both of the procedures, and are thus declared. Look this over carefully as it is very important to understand this concept of LOCAL and GLOBAL declarations. It may seem confusing at first, but after you work with it awhile it becomes almost second nature.

I will not take the time to explain all the formulas used, but they will work and do what they are supposed to do. The FOR loop is also new. It's similar to that in BASIC but a NEXT is not present. In this case you DO the line following the FOR

statement until the integer, I, evaluates to the contents of the variable Y, which is the number of payments. Notice that PROCEDURE CALCULATE calls PROCEDURE POWER, and that the variables INTERESTPERIOD and NUMBEROFPAYMENTS calculated in PROCEDURE CALCULATE are also used in PROCEDURE POWER. It's sort of an around in a circle type of situation, until the answer is evaluated. Again, don't worry about the formulas, it gets confusing. I just choose this example to expose you to these new concepts.

PROCEDURE DONE we have seen before. We are also familar with most of PROCEDURE PRINTANSWER, the part that causes output to console. This procedure will output the calculations to the console, and then duplicate to the printer with the addition of the input data so we can later make sense of the printout. The first new statement is REWRITE(PRINTER); which opens the LP: device driver for output. The rest is simply an output to the printer as previously described. To close the device driver buffer and be sure no characters are lost, again, the REWRITE(PRINTER); is used, and that brings us to the end of that procedure.

The PROGRAM BODY is next, and is all logical flow, nothing new, so it will not be discussed. Last month I printed a UCSD Pascal program. I have two Tiny Pascal programs I wanted to print this month at the end of this installment, but space does not permit that. I do not want to take over REMark! I'll save that for next month, and present you with more TOOLS of the Pascal language. Try these examples out, and try to write some of your own. See you next month.

EOF

Configure your FORTH for n-Drives

by: Glen B. Haydon
Haydon Enterprises
Box 429 Route
La Honda, CA 94020

I find the HUG enhancements to the fig-FORTH Model very good, but I find one very necessary routine missing.

When the H-47 drives first came out, I already had other versions of fig-FORTH operating and quickly wrote a program which allowed me to access the 8 inch drives as well as the 5 inch drives. Actually, I wrote the program the night before I went down to the local Heathkit Center without ever seeing the H-47. Needless to say, I was delighted to have the program work immediately. Utilizing this routine, I now have on line 2764K bytes of program and data space when the extended density formatting of the 8 inch drives is used (actually slightly less because of the dedicated nature of track 0 on each disk). Furthermore, I can quickly configure the 8 inch drives to single sided single density format which is the form of the IBM 3740 standard used on many, many systems or any other desired disk format. This makes the FORTH screens and their programs more widely portable and versatile. A further example of the potential portability of FORTH is that the screens presented here were read by the HUG implementation of fig-FORTH and my 'CONFIGURE' definition was immediately available with that implementation too. The screens presented here have been edited slightly for this presentation, but the program remains the same as that written the night before I first saw an H-47.

In typical FORTH programming style, the business part of the routine is on the last screen, where the FORTH word 'CONFIGURE' is defined. The earliest screens provide the utilities for 'CONFIGURE' to operate. Several arrays and words for accessing the appropriate data are also defined. Definitions of DR0, DR1 are extended through DR4. A new word 'T&SCAL' is defined using the word 'T&SCALC' in the assembly listing as a model. A jump to the new 'T&SCAL' is then patched on top of the old 'T&SCALC'. Finally, the word' CONFIGURE' is defined.

The original word 'DENSITY' is used but now has the possibility of having one of 7 different values-0 to 6. Some of the other hooks to CP/M are used as defined in the assembly listing. A discussion of the program follows.

The FORTH word, 'MAX.DRV' is simply a constant containing the current maximum number of drives. Next an array named 'DEN' is defined and initialized with values for a full system of 5 drives with the density of each which can be modified when the system is configured. The word 'DR-DEN' takes a drive number from the stack and returns the value of the current density on that drive. Next an array named

'SEC/DR' is defined and initialized with the number of sectors on a drive according to the options available. The word 'SPDRV' finds the number of sectors on a drive according to the current value of 'DENSITY'. Then an array 'SEC/TR` is defined with the number of sectors per track for each of the possible density options. The work 'SPT' finds the sectors per track for the current value of 'DENSITY'. We are then ready to define 'T&SCAL'.

As stated, 'T&SCAL' is modeled on 'T&SCALC' as given in the assembly listing. Based on the block number on the top of the stack, the proper drive, sector, and track are calculated and assigned to the respective variables for the actual writing or reading of a sector to or from a disk exactly as is done in the original 'T&SCALC'. The only difference is that the increased number of drives possible and the various options of format on each of the drives are considered. The current options are defined with the routine run by 'CONFIGURE' and referenced on each disk access. The system can be reconfigured quickly and easily by reexecuting 'CONFIGURE'.

Though the routine defined by the word 'CONFIGURE' is long, it is straight forward. It can be used when CP/M is booted up on an 8 inch drive by reassigning the order of the drives properly, and must be done before attempting to read any screens. For those who have the H-37 or H67, it should be a simple matter to redefine the options in the arrays 'SEC/DR' and 'SEC/TR' and adjusting the menu in 'CONFIGURE' accordingly, or if you have all types of drives, by extending the respective arrays and menu.

Finally, once one has configured his system for the number of disks available and the respective formats most commonly used, he can invoke the 'SAVE' extension to fig-FORTH provided in the HUG implementation 'SAVE'. This word requires that the drive number on which the binary image of FORTH is to be saved is on the top of the stack. In order to save these new screens as a CP/M COM file, 'SAVE` also requires that certain values be saved in low memory for start up. The technique for this is described in the HUG CP/M file FORTH.DOC - near the end. The actual changes will vary with ones particular interests. Limiting the 'SAVE' to images only through 'EDITOR' or 'ASSEMBLER' is for me, most inconvenient. A short outline of the changes I have made follows.

First I modified CP/M to load FORTH automatically after a fresh boot. Before using the word 'SAVE', I replace the user variables stored in low memory, by doing the reverse of the move done by 'COLD' as listed in the assembly source. Then I modified the 'SAVE' code explicitly to delete the existing FORTH.COM and saving the current image of FORTH as FORTH.COM rather than FORTH1.COM. These modifications are incorporated in a new FORTH word 'SAVE-IMAGE'. With this routine I can modify my working version of FORTH at any time. It saves a considerable amount of time when debugging new programs. Once a new definition works, the image of FORTH including it is saved. Then little time is lost with errors causing the system to crash. Rather than reloading the screens all over, an object module is immediately available which will reboot in seconds. One word of warning: DO NOT DO THIS TO YOUR DISTRIBUTION DISK !

In conclusion, the screens described work well and make possible a great extension of the on line storage area. I have implemented relatively large data bases in FORTH such as one with 5012 records of 128 bytes each and several 10240 byte images of keys to the records all of which are stored on a single 8 inch disk. At the same time it is possible to reconfigure a drive to read the standard single sided density disks which makes for portability of programs among many systems.

```
 0   CR ." SCREEN 80: CONFIGURE  -  1 of 4 "
 1
 2   3 CONSTANT MAX-DRV       ( value changed with CONFIGURE)
 3
 4   0 VARIABLE DEN   0 , 0, 1 , 1,
 5      ( DEN is an array of the values of DENSITY  on each drive )
 6
 7   : DR-DEN     (  drive number --- value for drive's DENSITY )
 8       2 * DEN + @ ;
 9
10   800 VARIABLE SEC/DR   2000 , 4000 , 4000 , 8000 , 4928 , 9856 ,
11      ( SEC/DR is an array of the sectors for each drive DENSITY )
12
13   : SPDRV  ( --- sectors per drive at current DENSITY)
14      DENSITY 2 6 MIN 2 8 SEC/DR + @ ;
15
```

```
 0   CR ." SCREEN 81: CONFIGURE  -  2 OF 4 "
 1
 2   20 VARIABLE SEC/TR   26 , 26 , 52 , 52 , 64 , 64 ,
 3        ( an array of SEC/TR for possible DENSITY(s)
 4
 5   : SPT  ( --- sectors per track for the current DENSITY )
 6     DENSITY @ 6 MIN 2 * SEC/TR + @ :
 7
 8   : SET-DRX   DR-DEN DENSITY !   SPDRV OFFSET +!   ;
 9     ( a utility for the following definitions )
10
11   : DR1    DR0  0 SET-DRX ;
12   : DR2    DR1  1 SET-DRX ;
13   : DR3    DR2  2 SET-DRX ;
14   : DR4    DR3  3 SET-DRX ;
15
```

```
 0   CR ." SCREEN 82: CONFIGURE  -  3 of 4
 1
 2   : T&SCAL ( rewrite of T&SCALC for other DENSITY(s)
 3      MAX-DRV 0 DO      ( do for the current MAX-DRV )
 4        DUP I DR-DEN DENSITY !  ( for each current drive )
 5        SPDRV /MOD   ( see if it is in a higher drive )
 6          IF   DROP  SPDRV -    I 1+ MAX-DRV =
 7            IF   DROP  ENDIF   ( loop if the drive exits )
 8          ELSE  I DRIVE !  SET-DRIVE  ( drive found; set it up )
 9            SPT /MOD TRACK !  1 + SEC !    DROP   LEAVE
10          ENDIF          ( leave the do-loop  )
11      LOOP ;
12
13   ' T&SCAL CFA ' T&SCALC !   ' ;S CFA ' T&SCALC 2 + !    ;S
14      ( patch the old T&SCALC with the new T&SCAL )
15
```

```
 0   CR ." SCREEN 83: CONFIGURE  -  4 of 4 "
 1   : CONFIGURE   27 EMIT 69 EMIT  ( clear display )  CR CR
 2   ." THIS VERSION IS CONFIGURED FOR -- " MAX-DRV .
 3    ."  -- DRIVES "
 4   CR CR ." WITH 'DENSITIES' OF " MAX-DRV 0 DO I DR-DEN
 5       2 SPACES   . LOOP CR CR
 6   ." 0 - 5-Single " CR ." 1 - 8-S Density S Sided " CR
 7   ." 2 - 8-S Density D Sided " CR ." 3 - 8-D Density S Sided " CR
 8   ." 4 - 8-D Density D Sided " CR ." 5 - 8-Ext Density S Sided "
 9   CR ." 6 - 8-Ext Density D Sided " CR CR
10   ." THIS WILL ALLOW YOU TO RECONFIGURE YOUR DRIVE DENSITIES "
11   CR ." AS WELL AS THE NUMBER OF DRIVES IN USE "
12   CR CR ." ENTER THE MAXIMUM NUMBER OF DRIVES  --  "
13   QUERY    32 WORD HERE NUMBER  DROP ' MSX-DRV ! CR
14   MAX-DRV 0 DO ." DRIVE " I . ." ? " QUERY INTERPRET
15      I 2 * DEN + ! ( store in DEN array ) CR LOOP ; ;S
                                                      EOF
```

Vectored from page 5

improvement was found for two real-life programs. One was a color graphics program by Bill Garner in MT+ Pascal, and the other was my modeling program. All of these programs are number-crunchers.

The chip actually does arithmetic much faster. A square root takes 1/30th of the time and trig functions are about 15 times as fast. Multiplications are twice as fast and divisions are 5 times as fast. However, there is enough overhead and format conversion that the whole program is often only tiwce as fast. One possible drawback is that the chip does not support double precision math.

One reason for buying the H-8 was that these "soup-up" parts were available now. However, it won't be long before an arithmetic processor board is available for the H-89 and there are a number of articles in BUSS and H-8 SCOOP on how to make the H-89's cpu run at 4 mHZ. EOF

MAKING SENSE OF MAKEBIOS

One of the nice things about the newest Heath version of CP/M is the ability to assemble the BIOS for your system, with what you need included and what you don't need left out. The MAKEBIOS program included with CP/M makes assembling the BIOS fairly easy, but the procedure is hard to understand for some, and it is particularly hard to do on 5.25 inch single sided single density drives. This article will present a simple procedure for assembling the BIOS that can be done on two drives of any type without disk swaping, and on one drive with less swaping than with the standard procedure. I assume that you have been reading the REMark series on CP/M and/or the CP/M documentation and have some idea of what the CP/M programs MOVCPM and SYSGEN do.

A New MAKEBIOS.SUB

One of the most confusing things about the BIOS assembly procedure is the SUBMIT file MAKEBIOS.SUB, because it is hard to decide which drives to specify as parameters on the command line, and difficult to figure out what goes on which disk. The following is a replacement SUBMIT file that requires no parameters. Use an editor to enter this file, and call it MAKE.SUB (to set it apart from the original).

```
MAKEBIOS B:1 B:
ASM BIOS.BBZ
REN B:BIOS.HX0=BIOS.HEX
MAKEBIOS B:2 B:
ASM BIOS.BBZ
REN B:BIOS.HX1=BIOS.HEX
STAT B:BIOS.ASM $R/W
ERA B:BIOS.ASM
PREL B:BIOS B:
MAKEBIOS B:3 B:
STAT B:BIOS.SYS $DIR
```

Assembling the BIOS

To assemble the BIOS, you will need to prepare two disks. The first is a bootable system disk containing the following files: ASM.COM, MAKEBIOS.COM, our MAKE.SUB, PIP.COM, PREL.COM, STAT.COM, and SUBMIT.COM. Any other files that you may need later, such as SYSGEN.COM, can also be put on the disk, as long as there is at least 1k free space on the disk. Enter STAT to see how much space there is. The easiest way to make this disk is to DUPlicate your regular system disk, delete any files not needed to make room, and PIP the files you do need to it. If you have only one drive, use FORMAT to make a new disk and SYSGEN to make it a system disk, then copy the needed files to it. The second disk you will need should have only BIOS.ASM on it. Do not use your original BIOS.ASM disk because BIOS.ASM will be deleted during the procedure. This disk does not need to be bootable.

After you have prepared the two disks, boot up on the first one and place the second one in drive B:. If you have only one drive, you will be instructed to insert the second disk (disk B) in drive A: when it is needed, and to replace the first disk when it is needed. To start the procedure, enter SUBMIT MAKE and hit RETURN. If you have two drives, you can go take a coffee break at this point because the procedure will take a while. Single drive users will have to stay by the computer to swap disks. When the procedure is finished, the new BIOS.SYS will be present on disk B, ready for installation.

Installing the New BIOS

The first disk you should install the new BIOS on is the system disk you used for assembling it, to test it. First, copy MOVCPMnn (use the appropriate MOVCPM program, such as MOVCPM17.COM for an H17 disk), SYSGEN.COM, CONFIGUR.COM, and FORMAT.COM to the disk. Delete any files used in the first step (except STAT or PIP) to make room. Boot up on the disk and delete the old BIOS.SYS with the following commands:

```
STAT BIOS.SYS $R/W
ERA BIOS.SYS
```

Then copy the new BIOS.SYS (from disk B) to the system disk and enter

```
MOVCPMnn * A:
```

Again, you would actually enter MOVCPM17 for an H17 disk, etc. When MOVCPM is finished, run SYSGEN. When it asks for a source drive, hit RETURN, and when it asks for a destination, type A and then hit RETURN. When it prompts for a destination again, reset the system and re-boot. CONFIGUR will run automatically as it always does when you make a new system with MOVCPM.

You can use this disk to re-SYSGEN your other system disks if you wish. Place the disk you wish to SYSGEN in drive B: (if you have two drives) and enter SYSGEN. When it asks for a source, enter A and hit RETURN. When it asks "COPY BIOS.SYS?", type Y. When it asks for a destination, enter B if you have two drives, and hit RETURN. If you have one drive, enter A, insert the disk to be sysgened in the drive, and hit RETURN. You can sysgen as many disks as you wish, and when you are

finished replace the system disk you started with (if you removed it) and hit RETURN in response to the destination prompt. Keep in mind that the new BIOS will be larger than the old one if you re-assembled it for two drive types, so you must have space for it on the disk you are sysgening. Use STAT BIOS.SYS to see how big it is, and STAT B:BIOS.SYS to see how big the old one on other disks is.

Since the purpose of re-assembling the BIOS is usually to allow a new type of drive to be used, you probably want to make a system disk for the new drive. For example, suppose you had an H89 with one H17 type (hard sector) drive and you added the H37 controller and one or more new drives. Let's say that up to this point you have been using CP/M on the hard sector drive and you want to make a soft sector system disk. Use the system disk that you used to make the new BIOS. Boot up on it, insert a new disk into the soft sector drive and FORMAT it. Remember that when you boot from a hard sector drive, the soft sector drives are labeled D:, E:, and F:, for units 0, 1, and 2. After the new disk is formatted, run MOVCPM. Use the appropriate MOVCPM for the new drive type (MOVCPM37 in our example). Then run SYSGEN and hit RETURN when it prompts for a source. When it asks for a destination, enter the appropriate drive letter and hit RETURN. Hit RETURN again when it asks for another destination. Then use PIP to copy all of the files (including the BIOS) to the new disk as follows:

```
PIP D:=*.*[R]
```

When this is done, you can reset the system and boot up on the new drive. It can be used to make more system disks of the same type.

PS:

SDUMP Patches

The following patchs are improvements for the SDUMP program on HUG disk 885-1213. The first patch is for those who do not have MAC.COM and cannot re-assemble SDUMP. This patch causes sector patches to be made as soon as you enter WRITE instead of when you access another sector or exit SDUMP. Make the patch with DDT as follows (your entries are in **bold print**).

```
DDT SDUMP.COM
NEXT  PC
1280 0100
-L554
  0554  LHLD 0001
  0557  LXI  D,0027
  055A  DAD  D
  055B  DCX  H
  055C  DCX  H
  055D  DCX  H
  055E  SHLD 0562
  0561  CALL 0000
  0564  JMP  071B
  0567  CALL 0D42
  056A  LDA  11BB
-A557
  0557  LXI D,27
  055A  DAD D
  055B  MVI C,1
  055D  NOP
-^C   (control-C struck)
>SAVE 18 SDUMP.COM
```

Make sure that the data shown by the L command matches that shown above. If it does not, do not make the patch. Our stock is being updated with later versions.

If you have MAC.COM you can make the above patch to the source code and re-assemble. At the label WRTDSK:, make the following change (shown in bold type):

```
WRTDSK: MVI     C,1     ;"DIRECTORY" WRITE
        CALLBIOS DWRITE ;WRITE BUFFER BACK
        JMP     ENDFIL  ;EXIT
```

This next patch corrects a problem in the buffer input function in MACRO.LIB. In the current version, if you enter something, delete part of it, then hit RETURN, the character count returned reflects the number of characters after the deletions, but the deleted characters are still in the buffer. For example, if you enter

```
=>TRACK 0 SECTOR 11
```

then press DELETE and hit RETURN, SDUMP displays sector 11, not sector 1 as it should. To correct the problem, make the following changes to MACRO.LIB, starting at the label INPUT.

```
INPUT   MACRO   ADDR,BUFLEN
        MVI     C,10
        IF      NOT NUL ADDR
        LXI     D,ADDR  ;;SET BUFFER ADDRE
        ENDIF
        PUSH    D       ;;SAVE BUFFER ADDR
        IF      NOT NUL BUFLEN
        MVI     A,BUFLEN ;;SET BUFFER LENG
        STAX    D
        ELSE
        MVI     A,127
        STAX    D       ;;SET BUFFER DEFAU
        ENDIF
        CALL    BDOS    ;;BDOS ENTRY
        POP     D       ;;RESTORE ADDRESS
        INX     D
        LDAX    D       ;;GET NO. OF CHARS
        INX     D       ;;MOVE TO STRING
        ADD     E       ;;ADD CNT TO ADDR
        MOV     E,A
        JNC     $+4     ;;NO CARRY
        INR     D       ;;ADD CARRY
```

Vectored to page 20

New HUG Software

885-1216 HUG CP/M BIOS for 2.2.03 $40.00
(This is a two disk set)

The HUG CP/M BIOS is a set of patches and programs that support 80 track and/or double sided disk operation (H17-4 drives) on Heath hard sector controllers (H88-1 or H-17) under Heath/Zenith CP/M version 2.2.03. It includes the following:

BIOS80(c) -- This is a set of modifications to the BIOS developed by Livingston Logic Labs, P. O. Box 5334, Pasadena CA 91107 and P. Swayne. These modifications allow the BIOS to support 80 track and/or double sided drives. It can also recognize, read and write on a single sided disk in a double sided drive; and recognize, read, but not write on a 40 track disk in an 80 track drive. You are supplied a complete BIOS with the modifications in place, in both source form and assembled form, ready to install in your system.

MOVCPM80 -- This is a set of patches for the MOVCPM17 program that allows you to make bootable 80 track and/or double sided disks. A SUBMIT file is provided that automatically installs the patches for you.

SETDRV -- With this program you can configure the BIOS for the type of drives you have, setting the number of tracks, number of sides, and the step rate for each drive. Changes can be made to memory only or to memory and the BIOS on the disk.

INIT80 -- This is a disk formatting program that can make 40 or 80 track, one or two sided disks. It formats a disk according to the way the drive was set up with SETDRV, and you can specify the drive to use in the command line (e.g. INIT80 C:). An optional media check is provided that not only finds bad places on the disk, but prints them out by track and sector.

DUP17 -- This program is included (.COM file only) so you can duplicate 80 track and/or double sided disks. For a complete description, see the HUG Disk Duplication Utilities, below.

SETTIME and FIXSTM -- The HUG Bios includes the screen clock modifications presented in REMark #22. SETTIME lets you enable or disable the clock, set the time, or make the clock run faster or slower. FIXSTM is used to configure the SETTIME program for your BIOS installation.

The HUG BIOS is the answer if you want to add higher capacity drives to your system without adding a new controller. It provides the same support for CP/M that the HUG SY: device driver (885-1095) provides for HDOS. For H8 computers (with HA-8-8 or HA-8-6) or H89 computers (with H-88-7 ROM set or ROMs included with Z-89-37) and 32k RAM.

885-1217 Disk Duplication Utilities $20.00

Now you can back up those expensive master program disks and all your disks with increased flexibility, using the HUG Disk Duplication Utilities. The following programs are included (these are CP/M programs).

DUP17 -- This program can duplicate 40 or 80 track, one or two sided hard sector 5.25 inch disks. The destination disks do not have to be pre-formatted. It can duplicate a disk in a one drive system, using as few swaps as possible because it copies as much as your memory allows in each pass rather than one track at a time or one program at a time. If you have three drives, it can make two copies at a time. It can copy HDOS or CP/M disks (CP/M is required to run DUP17) in three modes: copy only, copy with quick verify, and copy with full verify. It can also be used to verify only. Requires at least 32k of memory, and the source is provided. The HUG BIOS is required for 80 track and/or double sided copying.

DUP37 -- This program can duplicate 40 or 80 track, one or two sided soft sector 5.25 inch disks, in single, double, or extended density. The destination disks do not have to be pre-formatted. DUP37 has the same capabilities as DUP17, above, including one-drive duplicating and making two copies at a time. Requires CP/M, Z-89-37, and 48k of memory. The souce is included.

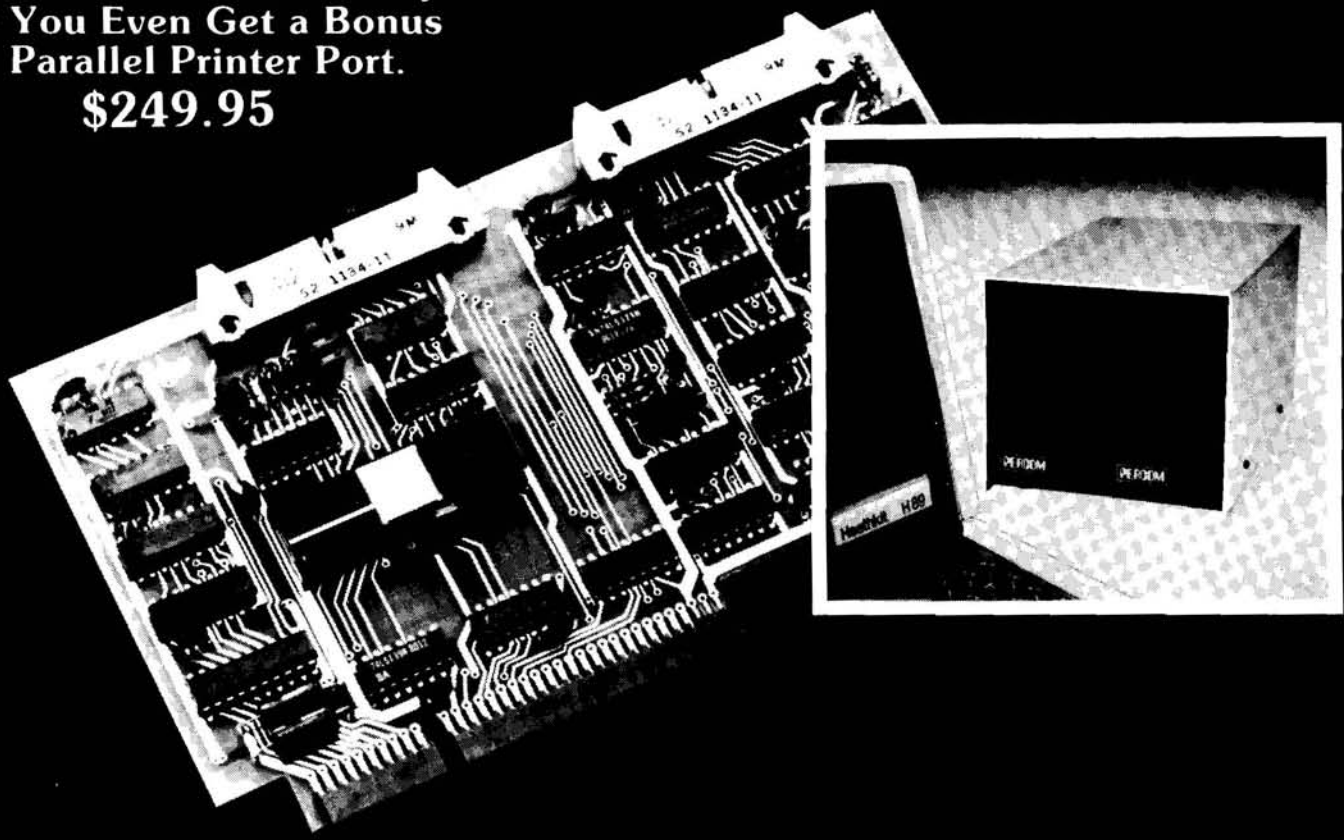
MAN37 -- We produced this program for our own use, but thought there might be a need for it elsewhere. It allows you to manufacture soft sector (40 track single side) copies from hard sector masters. If you sell programs on disk, and want to start distributing them in soft sector 5.25 inch format, you do not have to duplicate your hard sector master library when you use MAN37. It can copy HDOS or CP/M disks, and can make one or two copies at a time. Requires CP/M, an H89 with the H-88-1 and Z-89-37 controllers, at least two H-17-1 drives, and at least 32k of memory. Includes source.

885-1215 BASIC-E for CP/M 20.00

BASIC-E is a public domain BASIC P-code compiler written by the author of CBASIC. It is equivalent in computational power to Benton Harbor BASIC except that it has a kind of random file capability and is faster. It ran the test program in the

Percom's Double-Density Disk Controller... You Even Get a Bonus Parallel Printer Port.

$249.95



Expect more from Percom. You won't be disappointed.

Percom's *double-density* Z Controller for the H-89 is now available. Besides its many outstanding drive control features, the Z Controller includes a *bonus parallel port* that lets you directly connect your computer to a standard, off-the-shelf Epson MX-80, Okidata Microline 80 or other low-cost printer.

• Controls up to four single- or double-headed mini-disk drives. • Handles 35-, 40-, 77- and 80-track drives, and other standard track densities. • Formatted data storage capacity of 80-track diskettes is over 368 Kbytes. Forty-track diskettes store over 184 Kbytes. Capacities for other track densities are proportional. A Z system with four double-headed, 80-track drives provides almost 3 megabytes of on-line data. • The Z Controller co-resides with your H-89 disk drive controller. Your software can select either, and you don't have to move drives around when switching between systems. • The Z Controller includes Percom's proven digital data separator circuit and a dependable write-precompensation circuit. Expect reliable disk operation for a long, long time under 'Z' control. • The Percom Z Controller is priced at only $249.95, complete with HDOS-compatible disk drivers on diskette, internal interconnecting cable and comprehensive users manual.

System requirements – H-89 Computer with 24 Kbytes memory (min), Replacement ROM Kit H-88-7 and HDOS 2.0.

Add-On Z Drives for H-89, H-8 Computers

• Forty- and eighty-track densities in either 1- or 2-drive modules. • All drives are rated for single- and double-density operation. With a Z Controller, an 80-track drive can store over 364 Kbytes (formatted, one-side), a 40-track drive can store over 184 Kbytes. • Some models permit "flippy" storage, letting you flip a diskette and store files on the second side. • Z drives are fully tested, including a 48-hour operating burn-in to prevent shipment of drives with latent defects. • Assembled and tested one-drive units from only $399, two-drive units from only $795.

System requirements – H-89 or H-8 computer with 16-Kbyte RAM, Heath first-drive floppy disk system, HDOS and drives interconnecting cable. (Two-drive interconnecting cable optionally available from Percom)

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

Watch for announcement of 'Z' CP/M.

PERCOM

PERCOM DATA COMPANY, INC.
11220 PAGEMILL RD. DALLAS, TX 75243
(214) 340-7081

Toll-Free Order Number: **1-800-527-1222**

Yes . . . I'd like to know more about Percom Z drives and the Z Controller. Rush me free literature.

Send to
PERCOM DATA COMPANY, Inc., Dept. 26-R01
11220 Pagemill Rd. Dallas, TX 75243

name

street

city state

zip phone number

MAIL TODAY!

SMASH THE '89's 64K RAM LIMIT!

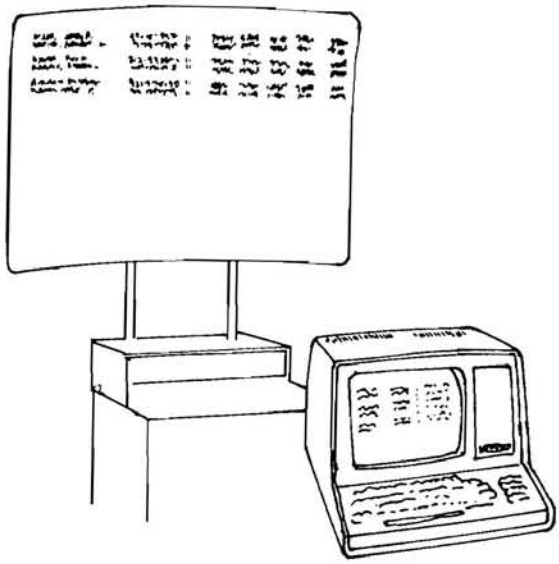
128K RAM BOARD — $595

order no. 77318

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

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

Delivery beginning March 15, 1982.



VIDEO OUTPUT — $79

order no. 77319

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

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

Delivery beginning March 15, 1982.

LOWER PRICE! 16K RAM — $125

order no. 77311

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

DOUBLE DENSITY DISK CONTROLLER — $595

order no. 77316

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

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

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

The package includes:

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

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

DOUBLE DENSITY SUBSYSTEMS

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

COMPLETE SYSTEMS

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

ORDERING INFORMATION

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

MAGNOLIA MICROSYSTEMS

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

Sunflower Software, Inc.

6304 WEST 76th STREET
PRAIRIE VILLAGE, KANSAS 66204

RICK KERBEL, *Pres.* **(913) 648-1333**

WORD PROCESSING

WORDPRO
A text editor/formatter heavy on features. When used with a "DIABLO", expect proportional word spacing, subscripts and exponents, etc. Takes the work out of word processing.

HDOS $59.95

SPELGUD/SUPERFOG
Flags misspelled words on your documents or correspondence. / Calculates suggested grade level of comprehension.

HDOS or CP/M $45.95

EDUCATIONAL

STATES
Teaches state recognition from graphic maps, capitols, nicknames, flowers, birds, trees and largest cities. An excellent geography learning aid.

DUALFORMAT $24.95

MATH
Aids the younger student in all four arithmetic functions. Will solve his school-work or generate arithmetic or word problems for him. Uses large graphic numerals.

DUALFORMAT $24.95

DATA MANAGEMENT

MAILMATE
Prints Labels, Letters or Lists. Merge Label data into your letters. A host of user definable options. Map driven output lets you determine output formats.

HDOS or CP/M $39.95

SYØ : NAMELIST
Single drive Name/Address/Phone manager. Operates in turn-key fashion. No experience necessary.

HDOS/MBASIC $19.95

COMPILERS/ TRANSLATORS

S-BASIC
Translates a "structured" Basic (B.H. or Microsoft) program into executable statements. An excellent introduction to other Structured Languages.

HDOS $49.95

CONTROL
A Data Entry/Base Language compiler. Supports data input editing and Random data retreival and update. Write business applications in a fraction of the time.

HDOS $99.95

UTILITIES

DUALPORT
Modifies HDOS to permit a "2nd" device to duplicate the I/O of your console terminal. Eleven functions give added control and security.

HDOS $39.95

MAGIC MENU
Converts any system to turn-key operation. Menu calls BASIC/ABS/ or any other program from command mode, and all programs automatically return to menu. Even executes multiple command strings. Menu headings are user defined.

HDOS $29.95

S.A.P.
5 utilities (CP/M, only the first 2) Search And Print, MSMAP, PHLAGS, DIRSORT, HMOD2

HDOS or CP/M $19.95

DISKSORT
Sorts any disk file (external) regardless of size. Select FIXED, VARIABLE, OR MULTIPLE RECORD records.

HDOS $29.95

APPLICATIONS

APPLIED STATISTICS
Complete statistics package for the engineer or student. All statistical functions including histogram.

DUALFORMAT/MBASIC $59.95

Executive Appointment SecretarY
Keeps track of appointments, meetings and schedules. Full screen display for each day. Ideal for Doctors, Lawyers, or anyone with a busy schedule.

HDOS or CP/M $39.95

SYØ:BILLING
Complete balance forward billing system designed for one drive. Accomodates 40 Accounts. Transactions are entered quickly and accurately. **HDOS/MBASIC $39.95**

Ordering Instructions: Sunflower Software can be seen and/or purchased at your nearest Heathkit Electronics Center. When ordering direct, include $1.50 for postage and handling. Kansas residents include sales tax. Specify operating system when applicable.

Attention Programmers: Sunflower Software is interested in high quality software for the H8/H89. Let us evaluate your original programs for our internationally marketed line.

Send For Free Catalog

DG IS Heath/Zenith

...with the F

H8 PRODUCTS

The Most Extensive Line of Hardware Support for The H8®

***DG-80/FP8**
Z80® based CPU including the powerful FP8 monitor—Both only $249.00. The acclaimed FP8 monitor package is now included with the DG-80 CPU!

***DG-64D/64K RAM Board**
Reliable, Low Power, High Capacity Bank-selectable RAM
Priced from $333.00 (0K) to $399.00 (64K)

***DG-64D5 64K/5 volt only RAM Board**
SUPER low power, Reliable, High Capacity, Bank Selectable RAM from $333.00 (0K) to $499.00 (64K)

***DG-32D/32K RAM Board**
Low cost, Dependable RAM for the H8 32K Version Only $179.00.

***DG-ADP4**
H17-4 MHz disk adaptor—$19.95

64K STATIC RAM–PROM BOARD

IF YOU STILL BELIEVE YOU WOULD RATHER USE STATIC MEMORY...OR IF YOU NEED A PROM/EPROM BOARD...
OR IF YOU NEED MORE ROOM ON YOUR H8 MOTHER BOARD...
OR IF YOU HAVE BEEN WAITING FOR STATIC MEMORY FOR THE H8 TO BECOME REASONABLY PRICED.
WE HAVE THE ANSWER—THE STATIC 64 RAM BOARD

FULLY STATIC.
SUPER LOW POWER—Power dissipation typically less than 4 watts (less than any memory board available for the Heath H8). Uses Single Supply 5-Volt memory devices.
CAPACITY—Up to 64K RAM OR Up to 64K EPROM (type 2716/2516) OR any combination of RAM and EPROM up to 64K.
SPEED—4MHz with NO wait states.
Compatible with the OLIVER ADVANCED ENGINEERING PROM Programmer Model #PP-2716.
Bank-select fully compatible with the DG-64D—Hardware selectable in 8K increments/software selectable in 16K increments.
Bank-select port addressable to any of the available 256 I/O ports using solderless jumpers.
On-board ROM-disable port for use with the DG-FP8 Monitor Package
Fully assembled and tested. Priced from $299.00 (0K) to $599.00 (64K). 16K Chip Sets: $125.00.

H8/H89 Software

Software Support for HDOS 2.0

*** Disk Management Utility Package**
Includes "Universal Dump", Intelligent Disk Dump Utility; "Universal Dup", Disk Copy Utility; and "BAD", Bad Disk Recovery Utility. $39.95 (Source: add $40.00)

*** Archive**
Space saving diskette back-up utility. For use with 5¼" or 8" disk systems. $39.95 (Source: add $40.00)

*** SYSCMD/plus**
A must for the serious HDOS user. Enhanced HDOS 2.0 system command processor provides extended commands and capabilities. $39.95 (Source: add $30.00)

*** Preload**
Support utility for systems using multiple, bank selectable memory boards. $29.95 (Source: add $20.00)

*** Advanced H17/H77 Driver**
Software driver for operation of double-sided and/or double-track double density drives with the H-17 and H-77 5¼" disk systems. Low cost alternative to high priced double density disk controllers. $19.95 (Source included).

Computer Support
lture Built-In

NEW SUPER 89

TURN YOUR H/Z-89 INTO A PROFESSIONAL QUALITY COMPUTER
HIGHER RELIABILITY NOW — MORE EXPANDABILITY FOR THE FUTURE

FEATURES	DG-SUPER 89	H/Z 89
USER MEMORY The DG Super 89 comes standard with 64K of user RAM "on-board". This configuration can be increased up to 256K of bank selectable/write protectable RAM.	64K-256K	16K-64K
Ø ORIGIN MODIFICATION No further modification, such as required on the H/Z-89, is necessary to operate in either a standard Heath/Zenith CP/M or HDOS 2.0 Operating System environment.	NOT NECESSARY	ADD-ON
CP/M-HDOS COMPATIBLE	YES	REQUIRES MODIFICATION
PERIPHERAL EXPANSION SLOTS DG Electronics has made provision in the design of the unit not only for compatibility with the standard factory expansion slots, but also for future expansion by doubling the number of available expansion slots on the unit to 6 instead of the standard 3.	6	3
ON-BOARD AMD9511 For those users who perform large amounts of arithmetic computations the DG Super 89 has provision on-board for use of the AMD 9511 arithmetic processor.	YES (PURCHASE SEPARATELY)	NO
CPU CLOCK FREQ. The CPU in the DG Super 89 operates at twice the speed of the standard H/Z-89.	4MHz+	2.048MHz
MULTI-USER CAPABILITY With up to 256K of bank selectable RAM on board the DG Super 89 offers the option of MULTI-USER CONFIGURATIONS of up to 4 users.	YES	NO
ENHANCED MONITOR DG Electronics has developed its own firmware monitor to allow the user greater flexibility and easier access to the advanced capabilities of the Z80 CPU.	YES	NO
REAL TIME CLOCK The DG Super 89 comes standard with an on-board real time clock.	YES	NO
PARITY CHECK ON RAM For those who are sticklers for accuracy, the DG Super 89 has parity check to make the user aware of errors occuring in the RAM during use.	YES	NO
SERIAL PORTS ON BOARD The DG Super 89 offers an additional serial I/O port for greater convenience and flexibility.	2	1

Now you can have all of the features in your H/Z-89 that you have always wanted. High speed and greater expandability are only the beginning of what our NEW DG SUPER 89 has to offer. DG Electronic Developments Co. has given the "89" capabilities of fast number crunching and data verification through parity. We have incorporated into the DG SUPER 89 such necessary items as 64K of user RAM, a powerful Keyboard monitor, and CP/M compatibility, items others require you to "add-on". Add to these features an extra serial port, a realtime clock, three more peripheral expansion slots, and multi-user capability and you have the computer that you really wanted to begin with; for a lot less than you would think.

Compatible with all currently available Heath/Zenith hardware devices.

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.

Local HUG News

Cleveland, OH has a second HUG which meets at the Heathkit Electronics Center at 28100 Chagrin Blvd. Cleveland, OH 44122 on the first Thursday of each month at 7:00 p.m.; contact Gerry Ciganko at (216) 291-1612 for further details.

HUG'EM (Eastern Massachusetts HUG) meets the third Wednesday of each month at 7:00 p.m. at the Heathkit Electronics Center at 165 Worcester Ave, Wellesley, MA 02181. For further information call (617) 237-1510.

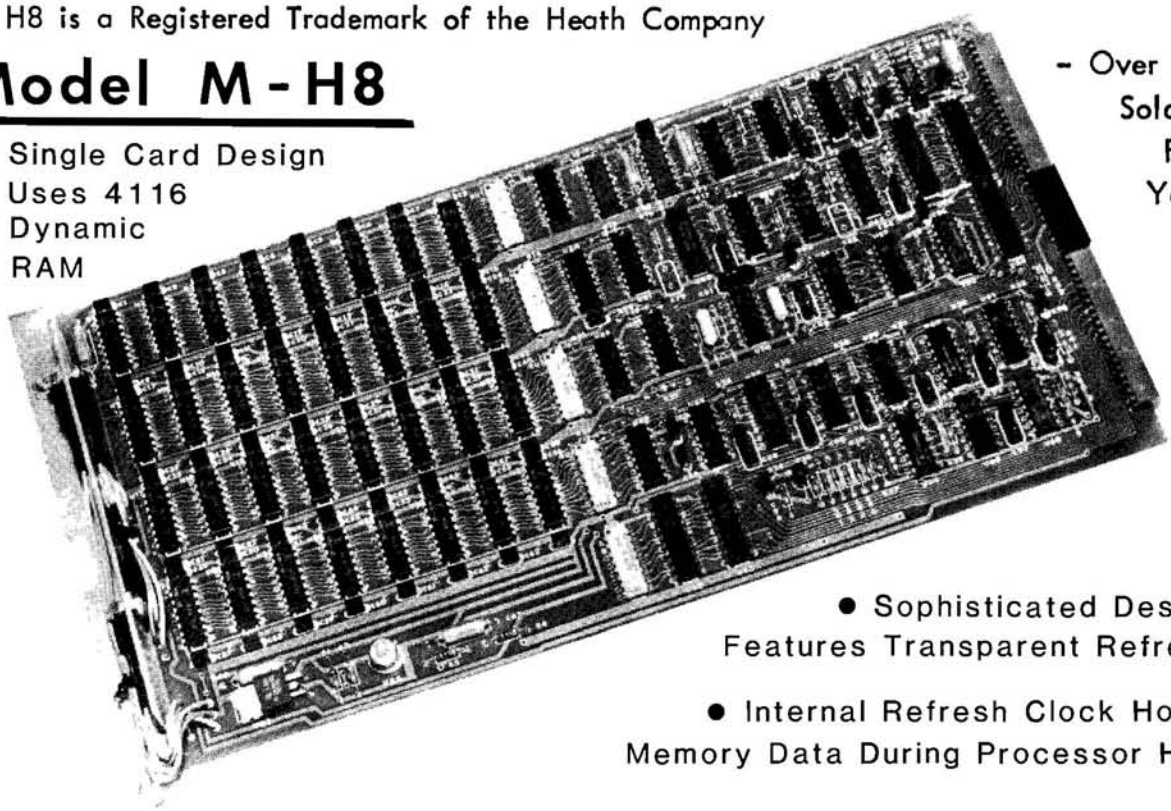
There is a HUG in Phoenix, AZ that meets the second Tuesday of each month at 7:00 p.m. at the Heathkit Electonics Center at 2727 W Indian School Rd Phoenix, AZ 85017. Interested individuals may contact John Grosberg at (602) 991-2515.

64K MEMORY for the H8*

* H8 is a Registered Trademark of the Heath Company

Model M-H8

- Single Card Design Uses 4116 Dynamic RAM



- Over 500 Sold in First Year!

- Sophisticated Design Features Transparent Refresh
- Internal Refresh Clock Holds Memory Data During Processor Halt

NEW PRICES:

Assembled - Without IC Sockets	Kit - IC Sockets Included	
$ 500	$ 415	64K
440	360	48K
380	305	32K
320	250	16K

16K Memory Expansion Kit - $ 60
(Memory Chips, Sockets, Capacitors)

Set of 8 Tested Memory Chips - $ 50

SPECIALS!

64K Kit -
Without Memory Chips
$ 225

64K Assembled -
Without Memory Chips
(Sockets Installed
for Memory Chips)
$ 300

Printed Circuit Board -
With Documentation
$ 50

Guaranteed To Work With Any Existing or New Heath H8 Products
— Upgrade Modification Instructions Will Be Issued As Required

Call or Write For Free Brochure • Call 714-830-2092 Ask For Bill Perry
Get On Our Mailing List For New H8 Product Announcements

TRIONYX ELECTRONICS, INC.

P.O. BOX 5131, SANTA ANA, CA 92704

The Zenith Microcomputer.

Compare the positions.

PROJECT MANAGER
Act as a group leader for our consumer oriented microcomputer products. BSEE/CS or equivalent and a minimum of 4 years experience in computer product design required; supervisory experience essential.

SENIOR HARDWARE DESIGN ENGINEER
Responsible for designing and developing hardware and systems for our microcomputer based products. BSEE or equivalent with strong digital circuit design background; exposure to software/firmware is desirable.

SOFTWARE CONSULTANT
Responsible for providing assistance, information and service for our customers concerning our microcomputer software products. BSCS or equivalent and demonstrated proficiency in operating systems languages and programming are required. A business background and excellent communications skills desirable.

SENIOR SOFTWARE ENGINEERS
Responsible for designing and implementing operating systems for microprocessor based computer systems including utility design and enhancements on existing products. Requires BSCS (or equivalent); plus at least 2 years experience with operating systems development (CP/M, HDOS desirable); and demonstrated assembly level language experience (8080, Z80).

Then compare the opportunities.

In terms of project challenge, state-of-the-art technology and ground floor potential, you'll find the opportunities at Zenith Data Systems incomparable. For example, take the Zenith Data Systems Z89 microcomputer . . . with dual Z80 microprocessor control, 64K of user RAM, 100K storage, built-in 12" diagonal CRT . . . only one of the new microcomputer, peripheral and software products we've developed.

Pure Innovation . . . Solid Zenith Quality
AND JUST THE BEGINNING!

Join us in the forefront of one of our most innovative phases. You'll find corporate wide impact and mobility, and an exceptionally stimulating project oriented environment. Compare it all . . . then call.

Kathy Kniola
616/982-3844

ZENITH | data systems

Hilltop Road / St. Joseph, MI 49085

A Wholly Owned Subsidiary of Zenith Radio Corporation

An Equal Opportunity Employer m/f/h

article "Speed on Heath Micros" in this issue in 23 minutes and 31 seconds, right up there with MBASIC. If you have CP/M but do not have a BASIC yet, BASIC-E will probably suffice until you can get MBASIC or CBASIC.

BASIC-E is in two parts, the compiler and the run-time interpreter. The compiler compiles your source file into a P-code file, and the interpreter runs it. Since all that you need to run a program is the interpreter and the P-code file, it saves disk space. You can protect software that you may wish to sell by distributing only the P-code file and the run-time interpreter. HUG BASIC-E includes a sample program and brief instructions, including a list of all key words in the language. BASIC-E is compatible with any standard (origin 0) version of CP/M 1.4 or higher.

SOFTWARE UPDATE

885-1207 TERM and HTOC $20.00

The TERM program on HUG disk 885-1207 has been replaced with a completely new version. It is now compatable with any version of CP/M 2.0 or higher, and does not need to be re-assembled when you change memory size. Features file load and dump, one key log-on, and simple operation. TERM includes the source and instructions. This disk also includes HTOC, a utility for copying files from HDOS to CP/M.

HUG Product List

Part Number	Description	Selling Price
	CASSETTE SOFTWARE (H8 and H88)	
885-1008	Volume I Documentation and Program Listings (some for H11)	$ 9.00
885-1009	Tape I Cassette	$ 7.00
885-1012	Tape II BASIC Cassette	$ 9.00
885-1013	Volume II Documentation and Program Listings	$ 12.00
885-1014	Tape II ASM Cassette H8 Only	$ 9.00
885-1015	Volume III Documentation and Program Listings	$ 12.00
885-1026	Tape III Cassette	$ 9.00
885-1036	Tape IV Cassette	$ 9.00
885-1037	Volume IV Documentation and Program Listings	$ 12.00
885-1039	WISE on Cassette H8 Only	$ 9.00
885-1057	Tape V Cassette	$ 9.00
885-1058	Volume V Documentation and Program Listings	$ 12.00
	HDOS SOFTWARE (H8/H17 or H89 -- 5-inch only)	
	MISCELLANEOUS COLLECTIONS	
885-1024	Disk I H8/H89	$ 18.00
885-1032	Disk V H8/H89	$ 18.00
885-1044	Disk VI H8/H89	$ 18.00
885-1064	Disk IX H8/H89	$ 18.00
885-1066	Disk X H8/H89	$ 18.00
885-1069	Disk XIII Misc H8/H89	$ 18.00
	GAMES	
885-1010	Adventure Disk H8/H89	$ 10.00
885-1029	Disk II Games 1 H8/H89	$ 18.00
885-1030	Disk III Games 2 H8/H89	$ 18.00
885-1031	Music 8 & 89 H8/H19 and H89	$ 20.00
885-1067	Disk XI Graphic Games .ABS and B H BASIC (H19/H89)	$ 18.00
885-1068	Graphic Games (H19/H89) *	$ 18.00
885-1088	Graphic Games (H19/H89) *	$ 20.00
885-1093	Dungeons and Dragons Game * Requires H89 or H8/H19	$ 20.00
885-1096	Action Games (H19/H89) *	$ 20.00
885-1103	Sea Battle Game (H19/H89)	$ 20.00
885-1111	HDOS MBASIC Graphic Games *	$ 20.00
885-1112	HDOS Graphic Games	$ 20.00
885-1113	HDOS Fast Action Games	$ 20.00
885-1114	Color Raiders and Goop (HA-8-3)	$ 20.00
	UTILITIES	
885-1019	Device Drivers (HDOS 1.6)	$ 10.00
885-1022	HUG Editor (ED) Disk H8/H89	$ 15.00
885-1025	Runoff Disk H8/H89	$ 35.00
885-1043	MODEM Heath to Heath H8/H89	$ 21.00
885-1050	M.C.S. Modem for H8/H89	$ 18.00
885-1060	Disk VII H8/H89 SUBMIT, CLIST, FDUMP, ABSDUMP, etc.	$ 18.00
885-1061	TMI Cassette to Disk H8 only	$ 18.00
885-1062	Disk VIII H8/H89 (2 disks) MEMTEST, DUP, DUMP, DSM	$ 25.00
885-1063	Floating Point Disk H8/H89	$ 18.00
885-1065	Fixed Point Package H8/H89	$ 18.00
885-1075	HDOS Support Package H8/H89	$ 60.00
885-1077	TXTCON/BASCON H8/H89	$ 18.00
885-1079	HDOS Page Editor	$ 25.00
885-1080	EDITX H8/H19/H89	$ 20.00
885-1082	Programs for Printers H8/H89	$ 20.00
885-1083	Disk XVI RECOVER, etc.	$ 20.00
885-1089	MACRO, CTOH, and misc Utilities	$ 20.00
885-1090	Misc. HDOS Utilities CCAT, HPLINK, AH, MBSORT, etc.	$ 20.00
885-1092	RDT Debugging Tool H8/H89	$ 30.00
885-1095	HUG SY: Device Driver HDOS 2.0	$ 30.00
885-1098	H8/HA-8-3 Color .ABS/.ASM	$ 20.00
885-1099	H8/HA-8-3 Color in Tiny Pascal	$ 20.00
885-1105	HDOS 2.0 Device Drivers MX-80, Paper Tiger, Clock, etc.	$ 20.00
	PROGRAMMING LANGUAGES	
885-1038	WISE on Disk H8/H89	$ 18.00
885-1042	PILOT H8/H89	$ 19.00
885-1059	FOCAL-8 H8/H89	$ 25.00
885-1078	HDOS Z80 Assembler	$ 25.00
885-1085	PILOT Documentation	$ 9.00
885-1086	Tiny Pascal H8/H89	$ 20.00
885-1094	HUG Fig-Forth H8/H89 2 Disks	$ 40.00
	BUSINESS, FINANCE AND EDUCATION	
885-1047	Stocks H8/H89	$ 18.00
885-1048	Personal Account H8/H89	$ 18.00

Vectored to page 31

SOFTWARE FOR H-11 USERS

by: Edward Judge
30 Autumn Drive
Northampton, MA 01060

I am writing for the few H-11 users out there who might want to know a little more about what software is available for the H-11/11-03/11-23. I have several PUBLIC DOMAIN softwares such as the NBS Pascal Compiler, SJSORT, a high-speed, general sorting utility, a C compiler (not complete as of yet-the new release is due "soon"), a FORTH system, a Structured Fortran preprocessor different from (and I feel better than) the RATFOR preprocessor, which I also have, including sources, and copying, spooling, xref, renumbering, and game programs, and others too numerous to mention. Also, John Runyon puts out a small periodical called "Computers-R-Digital", that is very informative for people who use DEC computers in any form whatsoever. Many of his subscribers use the H-11, and much good info is in the pages every month (approximately). John is the New York DECUS LUG Librarian, and can be reached for a sample copy of his magazine at: 39 Locustpoint Road, Locust NJ 07760.

In mentioning software for the LSI, I feel I should say something about the most important piece on the system, the monitor. RT11V4 is a very good o/s, and can be bought for $1100 without support, which means no updates. The range of utilities it offers is fantastic. Disc dumps, source compare for ASCII and binary files, updatable help file, disc copy, print, edit, and format programs. All have many options as to output, format, etc., and have reasonable write-ups and examples in the manuals. Some of the things they can do are amazing. One type of editor produces a command file that will do all the changes you made to another copy of the file you have edited. This is how most patches are made. Too much to go on about. Ask your nearest DEC rep to send you a copy of the latest RT11V4 Technical Specifications Manual. It will tell you all you need to know. Manuals can be gotten very cheaply at many universities. If they are a test site or a heavy user of DEC equipment, they can reprint manuals under some sort of agreement. I get many copies at the UMASS campus in Amherst.

On the commercial end, a very good word-processor is available called LEX-11. It has its own database, forms generation package, calculating facilities, graphics package, 90,000 word spelling dictionary for checking spelling, a desk diary, a mailing list package and integrated application program that allows it to be specially configured for any CRT terminal. It uses the "software interface" technique for terminal, peripherals, and printer, so it can be customized to work with any system.

Simile is a full featured database program with a powerful report generator that does everything. It can use files from other programs and can produce files for use with other programs. Command files can be used to execute long, involved or just tedious operations. It supports arithmetic field definitions and conditional commands, and error handling, along with full screen formatting for inputting data. It supports several terminals, including the VT-100, H-19, Z-19, and ADDS. It is one of the finest pieces of professional software I have ever seen. I use it for everything that HAS to be done and save my programming for things I enjoy. It will run on double density diskettes, but a small hard disc is better.

SPELL-11, a spelling checking program with a 30,000+ word dictionary, is very inexpensive. It has a menu driven format, making it easy to use, an excellent manual, and the ability to self expand the dictionary, or to use alternate dictionaries for any specialty vocabularies.

Although the public domain software has several sorts and a sorting utility, I feel I must mention a sort called ZORT available from a Canadian firm. This is without doubt, one of the most amazing pieces of software I have ever seen, and the fastest sort I have operated on my system. It has several interesting design features. It comes on a floppy with files ready to be incorporated into SYSLIB, FORLIB, DIBOL, BASIC, and the general utility version. It can be called as a final program in a chain as most others do, and it can be called as a subroutine which is not final in the program chain. Whenever it is called, however, it swaps out the program space of everything not needed to complete the sort, proceeds using maximum memory available and when finished, swaps the program space back into core. It is very fast because it absolutely minimizes I/O. On my 11/23, using RL01 hard discs, I sorted a 32K+ word (ASCII, average length 7 chars) dictionary in just under 5 minutes. Why it isn't better known is a mystery to me.

All of this software runs on TSX and TSXPlus.

Vectored to page 30

BUGGIN' HUG

To: Terry Jensen

In reference to your article in REMark 21, Page 23, I think I can shed some light on the "moving" VARPTR taken from the HDOS MBASIC Reference Manual, pages 7-13.

"All simple variables should be assigned values in a program before you call VARPTR for any array. Otherwise, allocation of a new simple variable will cause the addresses of all arrays to change."

Therefore, VARPTR is not just "jumping around", but rather following the new addresses assigned to arrays by MBASIC. VARPTR is the actual address of a variable at any point in time. The following table will indicate what to expect from VARPTR:

Type of simple variable	# of bytes the Address will change HDOS	CP/M
string	6	7
integer	5	6
single precision	7	8

For example, in the following program, the addresses printed will differ y 10 bytes, since 2 new integer variables are introduced between VARPTR's.

```
10 DEFINT A-Z
20 DIM U(10)
30 PRINT VARPTR(U(0))
40 K=1 : J=5
50 PRINT VARPTR(U(0))
60 END
```

If the two statements at line #40 were moved to line #25, then the printed addresses would be the same. MBASIC only changes the addresses of arrays during execution, so that the VARPTR's of simple variables will always remain constant.

The reason you had to add line #1000 to the "Real-time" program of REMark 18, page 24, was because a "new" simple variable was added at line #100 (A$), which is between the VARPTR of line #2060 and the usage of the routine at line #1000 (the original #1000 X=USR0(0)). If both the simple variables, X and A$, were predefined before the "DEF USR0 = VARPTR (U(0))", then your new lines of #1000 and #1005 would not have been needed. In the "POKE" program, the address is offset by 5 when using "Q=USR0" because "Q" is a "new" simple variable. This is consistant with the above table for integer variables under HDOS.

I would appreciate it if you could pass on to your readership that GENERIC SOFTWARE also has Data Base Management Software available. Current product offerings include:

(1) CARD-IT: a credit card management program

(2) MAIL-IT: a comprehensive name/address/phone # database system

For more information contact:

Dave Powers
GENERIC SOFTWARE
PO Box 1154
Troy, MI 48099
(313) 879-1115

Also to Terry:

You are to be congratulated on a fine job, both on the MicroNet HUG board, and also on your article in REMark issue 23 on Data Base Management Systems.

Please don't forget about one unique data base program which has its "roots" in HDOS! ROOTS89, the HDOS version of our genealogy program, is available through Softstuff as SF9008, while ROOTS/M for CP/M systems is SF9108. ROOTS is a unique data base in that, like Visicalc, the data is always present in RAM. The program uses a high degree of data compression to allow over 1600 records in a 64K CP/M system. If you consider that each record can contain up to 60 characters each, sex, marriage and children links, indication of certainty about entries, and a user defined flag, you can judge how powerful the data base is. Through careful structure of the data base, ROOTS achieves a 10-1 compression of data.

Keep up the good work! Sincerely,

Howard Nurse
COMMSOFT
665 Maybell Avenue
Palo Alto, CA 94306
(415) 493-2184

Dear HUG,

We are starting to use MBASIC to organize some of our officer personnel records. Since we are still using a single sided 40 track single drive disk system, we have implemented a method to maximize free space on our program and data disks.

1. SET HDOS STAND-ALONE on all system disks (REMark 12, page 13).

2. SYSGEN a system disk and delete enough unnecessary files to free the 80 sectors required for MBASIC. Copy MBASIC onto this disk.

3. Initialize two disks, labeling one MBASIC PROGRAMS and the other MBASIC DATA FILES. Copy only the following files onto each disk.

   PIP.ABS
   SYSCMD.SYS

4. Using an editor (or HDOS) create the file MBASIC.DOC as follows:

** When instructed to "Please Replace Diskette in Drive SY0:"

   Replace the disk in the drive with a disk containing MBASIC. The disk will be mounted, MBASIC loaded, and the drive reset to allow insertion of another disk. **

5. Using SUBMIT (HUG 885-1060) create the .ABS file below:

```
TYPE MBASIC.DOC
RESET SY0:
MBASIC/F:5    [# of files optional]
RESET "SY0:" [Note difference from
              previous RESET]
```

6. Copy the files from steps 4 and 5 to each of the disks created in step 3.

NOTE: In step 5 you must use the HDOS Reset command. The normally preferred "Safe Disk Reset -- R.ABS" (REMark 12) will dismount and remount the same disk, rather than allowing the disk to be changed (when used by SUBMIT).

To use the system:

1. Boot (or reset) to the MBASIC MASTER disk. Run MBASIC.

2. Reset "SY0:", insert MBASIC PROGRAMS disk. LOAD desired program.

3. Reset "SY0:" [our programs do this for us!] and insert MBASIC DATA disk. Run the program.

If for any reason you must return to HDOS while either the -PROGRAM or -DATA disks are mounted, the MBASIC program on each disk will reset the drive, load and reset MBASIC to allow another disk to be inserted in the drive.

Since we use many different programs with the same datasets, this method has been very effective.

Sincerely,

Charles F. Santose
HQ 3/107th ACR
4630 Allen Road
Stow, OH 44224

Dear HUG,

Thanks for the "MBASIC to Machine Code Link Revisited" in REMark 21. I had long been planning to load all my favorite programs from an elegant menu as a mixture of .ABS and .BAS files and this article combined with the "Menu-Driven Demo Program" article of REMark 19 and the "Screen Formatting in BASIC" article of REMark 12 certainly eased the task.

As fully explained in REMark 21, the MBASIC to Machine Code program actually begins the loading of the machine code program with the stack at 042176A. While this is "safe" and usually caused no problems I did find that "SPACE PIRATES" from the Software Toolworks would not execute from a "menu" because of this. The fix was simple...just change the last data entry in line 2170 from &0200 to &0202. This caused all machine code programs to begin at the "normal" USRFWA of 042200A, casued no problems with any other programs that I tried, and best of all made it possible to load "SPACE PIRATES" from my menu.

Truly yours,

Joseph E. McGlone
155 Lynn Drive
Mansfield, OH 44906

Vectored from page 15

```
XRA     A       ;;CLEAR A
STAX    D       ;;TERMINATE ENTRY
ENDM
```

This patch causes the entry to be terminated with a null, which SDUMP will recognize as the end of the entry.

PS:

Dear HUG,

The following MBASIC routine will convert a Split Octal number to Decimal. This is useful for interfacing MBASIC USR functions with ASM routines.

```
 10  '    Convert.Bas
 20  '    G. Hawthorne 10-16-81
 30  '    Split-Octl To Decimal Conversion
 40  DIM A(16),B(16)
 50  FOR I=1 TO 8:READ V(I):NEXT I
 60  FOR I=1 TO 8:READ B$(I):NEXT I
 70  FOR I=1 TO 16:READ A(I):NEXT I
 80  '
 90  LINE INPUT "Split-Octal String: ";S1$
100  GOSUB 210
110  PRINT "Binary Equivalent : ";S2$
120  PRINT "Decimal Equivalent: ";S3$
130  END
140  '
150  '    Stand-Alone Subroutines
160  '
170  '
180  '    Build Binary String From Split-Octal String
190  '    Entry: S1$=6 Position Split Octal String
200  '    Exit : S2$=16 Bit Binary String
210  FOR I=1 TO 6
220  Z=VAL(MID$(S1$,I,1))
230  FOR J=1 TO 8
240  IF I=1 THEN C$(J)=RIGHT$(B$(J),2):IF Z=V(J) THEN S2$=S2$+C$(J):GOTO 270
250  IF I=4 THEN C$(J)=RIGHT$(B$(J),2):IF Z=V(J) THEN S2$=S2$+C$(J):GOTO 270
260  IF Z=V(J) THEN S2$=S2$=B$(J)
270  NEXT J
280  NEXT I
290  '
300  '    Build Decimal String From Binary String
310  '    Entry: S2$=16 Bit Binary String
320  '    Exit : S3$=Decimal String
330  FOR I=1 TO 16:B(I)=VAL(MID$(S2$,I,1)):NEXT I
340  FOR I=1 TO 16:C=C+(B(I)*A(I)):NEXT I
350  S3$=STR$(C)
360  RETURN
370  '
380  '    Legal Split-Octal Values
390  DATA 7,6,5,4,3,2,1,0
400  '    Legal Binary Values
410  DATA "111","110","101","100","011","010","000"
420  '    Legal Decimal Values
430  DATA 32768,16384,8192,4096,2048,1024,512,256,128,64,32,16,8,4,2,1,
```

Gary Hawthorne
c/o Heathkit Electronics
22504 Ventura Blvd.
Woodland Hills, CA 91364

Dear HUG,

Following is a contribution to "The Basic Idea" column in REMark together with a demonstration program using these User Defined Functions and the results that might be expected.

I hope that you will find this material new and of interest.

THE BASIC IDEA

INVERSE TRIG FUNCTIONS

The only inverse trig function that is available in Benton Harbor BASIC is the ATN (arctan) function that returns the angle (in radians) when the tangent is given. Here are some User Defined Functions that will return the angle (in degrees) when the SIN (arcsin), COSINE (arccos), COTANGENT (arccot), COSECANT (arccsc) or the SECANT (arcsec) are given.

```
00010 REM  ARCCOT
00020 DEF FN U(X)=(ATN(1/X))*57.29578
00030 REM
00040 REM  ARCSEC
00050 DEF FN V(X)=(ATN(SQR((X*X)-1)))*57.29578
00060 REM
00070 REM  ARCCSC
00080 DEF FN W(X)=(ATN(1/SQR((X*X)-1)))*57.29578
00090 REM
00100 REM  ARCCOS
00110 DEF FN V(X)=(ATN(SQR((1-(X*X))/(X*X))))*57.295878
00120 REM
00130 REM  ARCSIN
00140 DEF FN Z(X)=(ATN(1/SQR((1-(X*X))/(X*X))))*57.29578
```

```
00005 REM PROGRAM TO DEMONSTRATE ACRSIN, ARCCOS, ARCCOT, ARCCSC & ARCSEC
00010 REM  ARCCOT
00020 DEF FN U(X)=(ATN(1/X))*57.29578
00040 REM  ARCSEC
00050 DEF FN V(X)=(ATN(SQR((X*X)-1)))*57.29578
00070 REM  ARCCSC
00080 DEF FN W(X)=(ATN(1/SQR((X*X)-1)))*57.29578
00100 REM  ARCCOS
00110 DEF FN Y(X)=(ATN(SQR((1-(X*X))/(X*X))))*57.29578
00130 REM  ARCSIN
00140 DEF FN Z(X)=(ATN(1/SQR((1-(X*X))/(X*X))))*57.29578
00300 LINE INPUT "IS THE RATIO SIN, COS, COT, CSC OR SEC? ";A$
00310 PRINT "WHAT IS THE ";A$:INPUT A
00320 IF A$="SIN" THEN C=FN Z(A):GOTO 400
00330 IF A$="COS" THEN C=FN Y(A):GOTO 400
00340 IF A$="COT" THEN C=FN U(A):GOTO 400
00350 IF A$="CSC" THEN C=FN W(A):GOTO 400
00360 IF A$=:SEC" THEN C=FN V(A):GOTO 400
00370 PRINT "FAULTY PICK, TRY AGAIN":GOTO 300
00400 D=INT(C):M=(C-D)*60
00410 IF M>59.4 THEN D=D+1:M=0
00450 PRINT "ANGEL--> ";D;"DEG."LINT(M+5);"MINS."
00500 PRINT :GOTO 300
```

```
IS THE RATIO SIN, COS, COT, CSC, OR SEC?
SIN
WHAT IS THE SIN
 .866
ANGLE-->  60 DEG. 0 MINS

IS THE RATIO SIN, COS, COT, CSC, OR SEC?
COS
WHAT IS THE COS
 .5
ANGLE-->  60 DEG. 0 MINS.

IS THE RATIO SIN, COS, COT, CSC OR SEC?
COT
WHAT IS THE COT
 1.7321
ANGLE-->  30 DEG. 0 MINS.
```

```
IS THE RATIO SIN, COS, COT, CSC OR SEC?
CSC
WHAT IS THE CSC
 1.4142
ANGLE-->  45 DEG. 0 MINS.
```

Sincerely,

Fred Wunder
16 Valley Road
Valley Cottage, NY 10989

The MENU Program Revisited

By: Eric L. Pang
2501 Soldiers Home Rd. 17G
W. Lafayette, IN 47906

In this article, I would like to present a menu program that loads and runs MBASIC programs along with binary programs. This allows the user, beginner or expert, to easily run a program without having to do much typing or have any knowledge of the operating system. This program displays the menu on the CRT and highlights a selection in reverse video. Items are selected by directing a pointer which is moved as indicated by the arrows on the keypad. Other options include, replacing the diskette in SY0: (I have a 1-drive system), loading an MBASIC program but not executing it (by making selection and hitting CTRL-L), or exiting to HDOS (CTRL-E). These last 2 options are not displayed to the user however.

The listing is as follows. The remark statments should explain most of the program logic. Note that this program is rather large for a menu program, but by removing the remark statements, eliminating unnecessary blanks, and using multiple statement lines, the size of this program can be reduced by at least 30%.

```
100 REM
105 REM  MBASIC MENU PROGRAM TO  LINK TO ANY MBASIC PROGRAM OR XXX.ABS PROGRAM.
110 REM                BY:  E.L. PANG    VER 1.1a    25-SEP-81
115 REM
120 REM  THIS PROGRAM TYPES A MENU TO THE SCREEN AND USES A POINTER TO INDICATE
125 REM  SELECTION DESIRED.  MBASIC PROGRAMS WILL BE IMMEDIATELY LOADED AND EX-
130 REM  ECUTED.  IF THE PROGRAM ENDS WITH  'RUN "MENU.BAS"', THE  MENU ON THAT
135 REM  DISK WILL BE LOADED AND EXECUTED UPON THE END OF THAT PROGRAM. MACHINE
140 REM  LANGUAGE PROGRAMS CAN ALSO BE EXECUTED FROM THE 'MENU.BAS' BUT   THESE
145 REM  WILL EXIT TO HDOS AT THE END OF EXECUTION.
150 REM
155 REM  NOTE: TO LOAD A PROGRAM ONLY, MAKE SELECTION AND TYPE <CTRL-L>.
160 REM        TO EXIT TO HDOS, TYPE <CTRL-E>.
165 REM
170 REM  VARIABLES: A$ = ARRAY HOLDING FILE NAMES
175 REM             C$ = VARIABLE HOLDING MACHINE CODE PROGRAM NAME
180 REM             C1 = CURRENT COLUMN OF POINTER
185 REM             D1 = PREVIOUS COLUMN OF POINTER
190 REM              I = CURRENT VALUE OF POINTER
195 REM              J = PREVIOUS VALUE OF POINTER
200 REM                M = MAX NO. OF ENTRIES IN 1 COLUMN (WILL HOLD 15 FILES+RES)
205 REM              N = NUMBER OF ENTRIES INCLD. RESET "SY0:
210 REM             N1 = NUMBER OF ENTRIES IN 1ST COLUMN IF 2 COLUMNS USED
215 REM             Y$ = INPUT VALUE
220 REM             Y2 = INPUT VALUE/2
225 REM             U0 = ARRAY HOLDING MACHINE LANGUAGE LINK
230 REM
235 CLEAR 100:DIM A$(30):DEFINT A-N,Y                    :REM DEFINE ARRAYS
240 E$=CHR$(27)                                          :REM ESC CHARACTER
245 P$=E$+"p":Q$=E$+"q"                                  :REM REVERSE VIDEO
250 X5$=E$+"x"+"5":Y5$=E$+"y"+"5"                        :REM CURSOR OFF/ON
255 B$=" ":N=0:M=4                                       :REM BLANK CHR
260 DEF FN C$(X1,Y2)=E$+"Y"+CHR$(31+X1)+CHR$(31+Y2)      :REM DIRCET CURSOR ADDR
265 PRINT E$+"z":PRINT:PRINT:PRINT X5$                   :REM RESET/CLEAR SCREEN
270 REM
275 REM                              FILES ON DISK
280 REM
285 DATA "OTHELLO.BAS","CATCH.BAS","PIRATES.ABS","INVADERS.ABS"
290 DATA "-99"                                           :REM LAST DATA ITEM
295 REM
300 REM                           READ FILES INTO ARRAY
305 REM
310 RESTORE:FOR I=1 TO 50:READ A$(I):IF A$(I)="-99" THEN 315 ELSE N=N+1:NEXT I
315 N=N+1:A$(N)="REPLACE DISKETTE"                       :REM INCLUDE RESET SY0:
320 REM
325 REM                        IF LIST LONG, NEED 2 COLUMNS
```

```
330 REM
335 C1=32:IF N<M THEN N1=N:GOTO 360                          :REM CHECK IF NEED 2
340 N1=INT((N+1)/2):C1=12                                   :REM NO. ITEMS 1ST COL.
345 REM
350 REM                              PRINT OUT MENU
355 REM
360 PRINT FN C$(2,4)+"PROGRAMS ON THIS DISK:"
365 FOR I=1 TO N1:PRINT FN C$(3+I,C1)+B$+A$(I)+B$:NEXT I    :REM PRINT FILE NAMES
370 IF N1=N THEN 380                                        :REM CHECK IF 2 COLUMNS
375 I1=0:FOR I=N1+1 TO N:I1=I1+1:PRINT FN C$(3+I1,50)+B$+A$(I)+B$:NEXT I
380 PRINT
385 PRINT:PRINT TAB(14);"USE ARROWS ON KEY PAD TO MOVE UP OR DOWN MENU LIST."
390 PRINT TAB(25);"Hit <ENTER> to run selection."
395 REM
400 REM                          HIGHLIGHT 1ST ITEM ON MENU
405 REM
410 I=1:J=I:D=C1                                            :REM CURRENT/OLD POINTR
415 PRINT FN C$(3+I,C1-2)+">"                               :REM PRINT 'CURSOR'
420 PRINT FN C$(3+I,C1)+P$+B$+A$(I)+B$+Q$                   :REM ENTRY REVRS VIDEO
425 REM
430 REM                             LET USER PICK ITEM
435 REM
440 Y$=INPUT$(1)                                            :REM INPUT VALUE
445 IF ASC(Y$)=13 THEN 675                                  :REM GOT PROGRAM
450 IF ASC(Y$)= 5 THEN PRINT Y5$+E$+"E":PRINT:SYSTEM        :REM EXIT TO HDOS
455 IF ASC(Y$)=12 THEN PRINT Y5$+E$+"E":PRINT:LOAD A$(I)    :REM LOAD PROGRAM ONLY
460 REM
465 Y=VAL(Y$):IF Y=2 OR Y=4 OR Y=6 OR Y=8 THEN 470 ELSE 440
470 Y2=Y/2                                                  :REM GOT A GOOD VALUE
475 REM
480 J=I:D1=C1                                               :REM SAVE OLD POINTER
485 ON Y2 GOSUB 560,585,610,645                             :REM MOVE POINTER
490 REM
495 IF J>N1 THEN 505                                        :REM RESTORE OLD LINE
500 PRINT FN C$(3+J,50)+E$+"o"+FN C$(3+J,D1)+B$+A$(J)+B$:J=I:D1=C1:GOTO 515
505 PRINT FN C$(3+J-N1,47)+E$+"K"+FN C$(3+J-N1,D1)+B$+A$(J)+B$:J=I:D1=C1
510 REM
515 IF I>N1 THEN 530                                        :REM HIGHLIGHT NEW LINE
520 PRINT FN C$(3+I,C1-2)+">"
525 PRINT FN C$(3+I,C1)+P$+B$+A$(I)+B$+Q$:GOTO 440          :REM RETURN TO INPUT
530 PRINT FN C$(3+I-N1,C1-2)+">"
535 PRINT FN C$(3+I-N1,C1)+P$+B$+A$(I)+B$+Q$:GOTO 440       :REM RETURN TO INPUT
540 REM
545 REM                          MOVE POINTER ROUTINES
550 REM
555 REM                                                          MOVE DOWN
560 I=I+1:IF I>N THEN I=N:RETURN                            :REM ALREADY AT BOTTOM
565 IF I<>N1+1 THEN RETURN                                  :REM IN SAME COLUMN
570 C1=50:RETURN                                            :REM SWITCH COLUMN
575 REM
580 REM                                                      MOVE TO LEFT
585 IF N1=N THEN RETURN                                     :REM ONLY 1 COLUMN
590 IF C1=12 THEN RETURN                                    :REM IN SAME COLUMN
595 C1=12:I=I-N1:RETURN                                     :REM MOVE OPPOSTIE ITEM
600 REM
605 REM                                                      MOVE TO RIGHT
610 IF N1=N THEN RETURN                                     :REM ONLY 1 COLUMN
615 IF C1=50 THEN RETURN                                    :REM IN SAME COLUMN
620 C1=50:IF I<=N1-1 THEN I=I+N1:RETURN                     :REM MOVE OPPOSITE ITEM
625 IF 2*N1=N THEN I=I+N1 ELSE I=I+N1-1                     :REM IS THERE OPPOSITE?
630 RETURN
635 REM
640 REM                                                        MOVE UP
645 I=I-1:IF I<1 THEN I=1:RETURN                            :REM ALREADY AT TOP
650 IF I<>N1 THEN RETURN                                    :REM IN SAME COLUMN
655 C1=12:RETURN                                            :REM SWITCH COLUMN
660 REM
665 REM                       GOT SELECTION.  GO RUN IT
670 REM
675 PRINT Y5$+E$+"E":PRINT                                   :REM RESTORE SCREEN
```

```
680 ON I GOTO 690,690,695,695
685 RESET "SY0:":RUN "MENU.BAS"
690 RUN A$(I)                                          :RUN MBASIC PROGRAM
695 PRINT                                              :REM WARNING MESSAGE
700 PRINT "NOTE: On EXIT you will be in HDOS."
705 PRINT TAB(6);"Type 'MENU' to restart available program listing.":PRINT
710 C$="SY0:"+A$(I)+B$                                 :REM DON'T FORGET BLNK
720 REM
725 REM      MBASIC TO MACHINE CODE LINK. BY JWF/GK.  1980. REMARK 12:30.
730 REM
735 CLOSE:DIM U0%(12)                                  :REM CODE STARTS HERE
740 U0%(0)=&H3FE  :U0%(1)=&HEBC0:U0%(2)=&H237E         :REM POKE IN CODE
745 U0%(3)=&H235E :U0%(4)=&H8356:U0%(5)=&H7A4F:U0%(6)=&HCE   :U0%(7)=&HB47
750 U0%(8)=&HFE0A :U0%(9)=&HC020:U0%(10)=&H2AF:U0%(11)=&HFFEB
755 U0%(12)=&HC920:DEF USR0=VARPTR(U0%(0))
760 PRINT USR0(C$);" CANNOT BE EXECUTED."
765 END
```

Lines 680-695 may have to be rearranged depending on the number of programs to be displayed by the menu and the types of programs to be executed. The MBASIC to machine code link is a concise and elegant way to execute a program with an *.ABS extension; however, this technique can not be used if a parameter has to be entered along with the file name. For example, if the programs that could be run were

```
285 DATA "OTHELLO.BAS","SINK.BAS","PIRATES DELTA.IV","PIRATES CONVOY
290 DATA "-99"                                         :REM LAST DATA ITEM
```

one has the option to run the program "PIRATES" with file "DELTA.IV" or "CONVOY". In this case, the variable C$ is changed to

```
710 C$="SYSTEM;"+A$(I)                                 :REM EXIT TO HDOS/RUN
```

and the routine by D.M. Deck, "Using the HDOS Type-Ahead Buffer", REMark 19 pages 18-19) is used in place of the machine link routine (lines 720-7605), where

```
735 REM START OF ROUTINE - Add the final C/R to the string. 740 C$=C$+";":T7=0
```

would start off the routine with the rest of the code following. This technique is just as if the commands were typed from the keyboard, thus "SYSTEM" and the file to be run will be displayed on the screen.

Note that a "warning" message is issued if an *.ABS program is run. Most of my MBASIC programs end with a 'RUN "MENU.BAS"', so when the program is exited, the menu is redisplayed. Obviously this can not be done with *.ABS programs. I have included MBASIC and MENU.ABS on the disks which have *.ABS programs, thus one need only type 'MENU' in response to the HDOS prompt. MENU.ABS is just a renamed PROLOGUE.SYS, which loads MBASIC.ABS and runs MENU.BAS (e.g. see J.H. Gold, " Another Pointer to the Type-Ahead Buffer", REMark 13 pages 24-27). If the user should abort the program, such as hitting a CTRL-C, thus leaving the system in an "unknown" state; it would be simple enough to have him/her reset the computer and insert the appropriately marked "turnkey diskette" (such as the disk with the big red dot). Since MBASIC does not have to be on all the disks (as RESET "SY0:" is used), the disks may not have to be 'SYSGEN'ed; leaving quite a bit of free space on the disk. Depending on what type of manipulations you may want to do on the disks, it may be easier to SYSGEN the disk anyway, use HDOS STAND-ALONE (which is invoked when a RESET "SY0:" is executed), and eliminate "unnecessary" files such as the HDOS system files and SY.DVD as they are subsequently locked into memory. PIP.ABS could also be eliminated (to remove the flags, see S. Robbins, "HDOS 2.0 Modifications", BUSS 33 pages 3-4 [watch out for typo FE-->FA] or P. Swayne, "Loosing Weight With HDOS 2.0", REMark 19 pages 12-15), and one would have quite a bit of free space on the media.

Improvements to the HUG SY: Device Driver

This article presents improvements to the HUG SY: device driver for 5.25 inch hard sector disks and HDOS 2.0 (HUG part no. 885-1095). The improvements will accomplish the following:

1. The control byte at 040066A is properly set up so that the device driver will work on all H8's and H89's, old or new.

2. The driver will load itself permanently into memory (at the first use of it) even if it is the alternate disk device (DK:). This eliminates the need to LOAD DK: and prevents system crashes when you dismount the last DK: disk.

3. Separate timers are used for head select and de-select times. This allows head select to be permanently set at the value used in the original Heath driver for maximum performance, while the user can set head de-select as necessary to prevent head banging during MBASIC program loads and editor file loads. (The original HUG driver uses the same timer for both select and de-select timing. This seriously degrades performance if a long select time is set.)

These modifications are for the version of the driver sold by HUG, and may not be suitable for other versions. Our stock will eventually be updated with a version containing these modifications.

Most of these modifications are to the file MFDVD.ACM., and the rest are to DKH17.ASM. Use an editor to make the modifications, then re-assemble DKH17.ASM and DKH17I.ASM, combine the two output files with PIP to produce SY.DVD, then run DVDDKGEN to set up certain addresses within SY.DVD. See the file SYDVD.DOC on the distribution disk for more information on re-assembling the driver.

MFDVD.ACM PATCHES

The first modification is in the file MFDVD.ACM, below the label MFDVD. The area of the modification is listed below. Additions and/or changes are shown in **bold print.**

```
        XRA     A
        OUT     UP.FC           ;SET FILL CHARACTER

        LDA     MFIDLET
        STA     D.XITA          ;SET IDLE TIMEOUT VALUE FOR MOTOR
        MOV     L,A
        MVI     H,0
        SHLD    D.DLYMO         ;SET MOTOR, RESET SELECT TIMEOUT COUNTS

        LHLD    MFIODLY
        SHLD    D.SDPA          ;SET I/O DELAY VALUES

        IF      MBOOT-1

        LXI     H,.CTL2FL       ;INITIALIZE PORT BITS
        MOV     A,M
        ORI     CB2.CLI         ;KEEP CLOCK ENABLED
        ANI     -1-CB2.SID-CB2.SSI      ;CLEAR SIDE AND SINGLE STEP
        MOV     M,A             ;UPDATE CONTROL BYTE

        ELSE

        LXI     HL,MFCLOCK
        SHLD    .UIVEC+1        ;SET REPLACEMENT CLOCK ROUTINE

        LXI     HL,0
        SHLD    D.ERTS          ;CLEAR ERROR TRACK/SECTOR NUMBER

        LXI     HL,MFERR
        SHLD    D.ERRT+1        ;ERROR TRAP ROUTINE

        LHLD    S.SYSM          ;UPDATE SYSTEM FWA
        SHLD    S.RFWA
```

```
        LHLD    AIO.DTA         ;GET DEVICE TABLE ADDRESS
        LXI     D,DEV.RES       ;GET OFFSET TO RESIDENCY FLAG
        DAD     D
        MOV     A,M
        ORI     DR.PR           ;FLAG PERMANENTLY RESIDENT
        MOV     M,A

        ENDIF
```

Notice that the code to set up the replacement clock routine was simply moved, not changed. The next change is below the label MFSDP, just under the DI instruction.

```
        DI
        STA     HDCTL           ;SAVE SELECT CODE
        CALL    MFDVSEL         ;GO SELECT DRIVE, RETURN DEVICE DIFFERENCE
```

The next change is at the label MFXITL1.

```
MFXITL1 LDA     D.DLYHS         ;GET HEAD LOAD DELAY
        ANA     A               ;HEAD FULLY LOADED -
        JNZ     MFXITL1         ;- NO, LOOP

        LHLD    D.XITA          ;GET IDLE TIMEOUT VALUES
        SHLD    D.DLYMO         ;SAVE TIMEOUT COUNTS
        LDA     MFSIDLY         ;GET HEAD UNLOAD TIMEOUT VALUE
        STA     MFSIDL          ;SET IT
        CALL    MFTMOUT         ;GO PROCESS INITIAL TIMEOUT VALUES
```

The next changes are at the label MFTMOUT

```
MFTMOUT XRA     A
        LXI     HL,D.DLYHS      ;GET HEAD LOAD TIMEOUT COUNT LOC
        CMP     M               ;HEAD LOADED -
        JNZ     MFTMOS1         ;- NO, GO PROCESS HEAD LOAD TIMEOUT

        LDA     MFSIDL          ;GET SELECT IDLE TIMER
        ORA     A               ;HEAD UNLOADED?
        JNZ     MFTMOS2         ;NO, PROCESS HEAD IDLE TIMEOUT

        DCX     HL              ;GET MOTOR ON TIMEOUT COUNT LOC
        CMP     M               ;MOTOR TIMED OUT -
        RZ                      ;- YES, RETURN

        LDA     .TICCNT
        ANA     A               ;DECREMENT MOTOR TIMEOUT COUNT -
        RNZ                     ;- NO, RETURN

        DCR     M               ;DECREMENT MOTOR TIMEOUT COUNT
        RNZ                     ;MOTOR NOT TIMED OUT, RETURN

        JMP     MFDVOUT         ;GO RESET DISK MOTORS

MFTMOS1 DCR     M               ;DECREMENT HEAD LOAD TIMEOUT COUNT
        RNZ                     ;RESULT NOT ZERO, RETURN

        DCX     HL              ;SET MOTOR ON TIMEOUT COUNT LOC
        CMP     M               ;MOTOR TIMEOUT IN EFFECT (HEAD LOADED) -
        RZ                      ;- NO, RETURN
        LDA     HDCTL           ;THIS IS NOT DE-SELECT, KEEP HEAD ON
        JMP     MFDVSEL         ;SELECT DEVICE

MFTMOS2 DCR     A               ;DECREMENT HEAD IDLE TIMER
        STA     MFSIDL          ;UPDATE IT
        RNZ                     ;RESULT NOT ZERO, RETURN

        DCX     H               ;POINT TO MOTOR TIMER
        CMP     M               ;MOTOR STILL RUNNING?
        RZ                      ;NO, RETURN

*       JMP     MFDVSEL         ;GO DESELECT
```

The next change in MFDVD.ACM is at the label MFIDLET

```
MFIDLET DB      MFMIDLE*2+1
MFSIDLY DB      MFSIDLE+3/4+1
MFSIDL  DB      MFSIDLE+3/4+1

HDCTL   DB      0               ;HEAD CONTROL BITS

MFIODLY DB      MFONDLY+3/4,FMOFDLY+3/4
```

The final change in MFDVD.ACM is at the label MFSIDLE. Since setting a longer head deselect time actually improves performance instead of degrading it as before, we set the default timeout values to the maximum allowed.

```
        IF      MBOOT-1

MFSIDLE SET     1016            ;BOOT HEAD IDLE TIME IN MILLISECONDS

        ELSE

MFSIDLE SET     1016            ;DEFAULT HEAD IDLE TIME IN MILLISECONDS

        ENDIF
```

DKH17.ASM PATCHES

The next changes to improve the HUG SY: driver are in the file DKH17.ASM. The first two changes are for cosmetic reasons only. In the original driver, the SELECT SET option affected both head select and deselect times. In the modified one, it affects only deselect time, so I changed the option from SELECT to HEAD, to conform with using MOTOR for setting motor deselect time. The first of these changes is at the label HELP.

```
HELP    CALL    $TYPTX
        DB      NL,NL,'Set Options:',NL
        DB      NL,'    All units -',NL,NL
        DB      'MOTOR nnn      Motor idle time (sec)',NL
        DB      'HEAD nnnn      R/W Head idle time (ms)',NL
        DB      NL,'    Each unit -',NL,NL
        DB      'STEP nnn       Track step time (ms)',NL
        DB      'SIDES n        Number of sides',NL
        DB      '[40/80]TRK     Number of tracks per side',NL
        DB      ENL
```

The next modification is at the label OPTTAB.

```
OPTTAB  DW      OPTABE          End of Option Table
        DB      1               1 data byte

        DB      'HEL','P'+200Q,HELPI

        DB      'MOTO','R'+200Q,MOTORI

        DB      'HEA','D'+200Q,SELECTI

        DB      'STE','P'+200Q,STEPI
```

This next modification is the only one to DKH17.ASM that is absolutely necessary. It is below the label SELLP1.

```
        ADC     H               GET (ORIGINAL VALUE + 3)/4 + 1
        JC      INVALID         TOO BIG

        STA     MFSIDLY         STORE VALUE
        RET
```

This completes the modifications to the device driver. In the original driver, the SELECT option took effect immediately, but in the new one, the HEAD option only takes effect after you re-boot the system.

MODIFYING DUP

The DUP program (on HUG disk 885-1062) had trouble working with the original driver, and will absolutely not work with the new one, but the patch to fix it is minor. This patch will also make it work better with the original driver and will not affect operation with the Heath driver or older HDOS. First, you should add a definition for D.DLYHS in the EQU statements at the beginning of the program.

```
TICCCNT EQU     40033Q          CLOCK
D.DLYHS EQU     40244A          R/W HEAD LOAD TIMER
S$MOUNT EQU     41027A          MOUNT SAFETY SWITCH
S.MOUNT EQU     41032A          SWITCH FOR VERSIONS 1.6 AND LATER
```

Next, three lines of code must be added after the comment "WRITE A COMPLETE TRACK".

```
*
* WRITE A COMPLETE TRACK
*
RWAIT   LDA     D.DLYHS         READ FINISHED?
        ANA     A
        JNZ     RWAIT           IF NOT, WAIT FOR IT
        MVI     A,1             ALL WRITES ARE TO DRIVE 1
```

After this modification, DUP will run under the new device driver, but it will not copy 80 track or double sided disks. To make DUP copy those disks, additional patches are needed. First, delete the line DSKTKS EQU 40 near the beginning of the source. Then make these patches, the first of which goes just before the previous patch.

```
        MVI     B,256*10        ONE TRACK
        CALL    MREAD           PICK IT UP
        CC      RSS             NO GOOD - TRY READING SINGLE SECTORS
        LDA     OURTS           GET CURRENT TRACK
        ORA     A               TRACK 0?
        JNZ     RWAIT           IF NOT, CONTINUE
        CALL    SETTRK          FIND OUT HOW MANY TRACKS
*
* WRITE A COMPLETE TRACK
*
RWAIT   LDA     D.DLYHS         READ FINISHED?
```

The next patch is at the label TKDONE.

```
TKDONE  LHLD    OURTS           GET DISK POSITION
        INR     L               INCREMENT TRACK COUNT
        MVI     H,0             RESET SECTOR TO 0
        SHLD    OURTS
        LDA     DSKTKS          NO. TRACKS/DISK
        CMP     L               CHECK NEXT TRACK
        JZ      DONE            ALL DONE
```

The next patch is a few lines below the label TKCK0.

```
        LXI     B,256*10        READ TRACK
        CALL    MREAD           INTO IOBUF
        CC      RSS             TRY SINGLE SECTORS
        LDA     OURTS
        ORA     A               ON TRACK 0?
        JNZ     TKCK0A          NO
        CALL    SETTRK          FIND OUT HOW MANY TRACKS
TKCK0A  LHLD    OURTS           GET POSITION AGAIN
        SHLD    R.TT            INFORM HDOS
```

The next patch is below the label TKCK1.

```
        LHLD    OURTS           GET POSITION AGAIN
        INR     L               INCREMENT TRACK NO.
        SHLD    OURTS           SAVE IT
        LDA     DSKTKS          GET NO.OF TRACKS
        CMP     L               DONE?
```

```
        JNZ     TKCK0           NO.
        RET                     YEP!
```

Two of the patches above call a routine called SETTRK, which must be added to the program. A good place to add it is just before the routine RSS.

```
*
* FIND OUT NO. OF TRACKS ON SOURCE DISK, AND SET DSKTKS
*
SETTRK  LDA     IOBUF+VIDLOC+16 GET DISK TYPE
        MVI     B,40            ASSUME 40 TRACKS
        ORA     A               IS IT 40?
        JZ      STTKS           YES
        MVI     B,80            ASSUME 80 TRACKS
        CPI     1               TWO SIDE 40 TRACK DISK?
        JZ      STTKS           YES
        CPI     2               ONE SIDE 80 TRACK DISK?
        JZ      STTKS           YES
        MVI     B,160           ASSUME 160 TRACKS
        CPI     3               TWO SIDE 80 TRACK DISK?
        JZ      STTKS           YES
        MVI     B,40            BAD LABEL (PROBABLY CP/M DISK)
STTKS   MOV     A,B
        STA     DSKTKS          SET NO. OF TRACKS
*
* READ A FULL TRACK, ONE SECTOR AT A TIME
*
RSS     CALL    EFAST           MAKE ERRORS FAST
```

The final modification to DUP is to add a place to store the number of tracks on the disk. This should be done in the data area, at the end of the source.

```
*
* DATA AREAS
*
DSKTKS  DB      0               TRACKS PER DISK
X.DTS1  DS      2               HDOS TABLE SAVES
X.LPS1  DS      2
```

As you can see by examining the routine SETTRK, this version of DUP determines how many tracks are on a disk from the label. If the disk has no label (as in the case with CP/M or UCSD disks), it is assumed to have 40 tracks.

PS:

Local HUG News

Tim Davis of 3504 Casino St Fargo, ND 58102 is interested in contacting other Huggies in his area. If interested you may call Tim at (701) 235-8288 or write.

Joe Williams of Rt 3 Box 39A Hillsborough, NC 27278 reports that the Research Triangle Park (HUG-RTP) is forming in North Carolina. Individuals interested may contact Joe at (919) 732-6678 or Steve Lassiter at (919) 544-2777 to establish a convenient meeting time and location.

FWHUG (Ft. Worth HUG) meets the second and fourth Tuesday each month at the Heathkit Electronics Center at 6825A Green Oakes Road Ft. Worth, TX 76116. No time was given for meetings; please call (817) 737-8822 for full information.

Vectored from page 18

I will copy the public domain stuff for $35.00 and a box of 8" floppies, or if someone wants to send an RL01 disc, I'll load it for $20.00 plus postage (RL02 soon). Whatever is sent must be in a good mailer or I'll send it back unopened. The writeups are available from DECUS or commercially, as is the case with "C". Many have documentation on the disc. I include a wide assortment on the 10 discs: games (Dungeons, Adventure, Super Star Trek, Life), utilities (sorts, statistics, spoolers, xref's, etc.), languages (FORTH, APL, PASCAL, C), so I hope to please most people. If you don't like some of it, someone else with something you like might want to swap. That's the name of the game.

Vectored from page 17

885-1049	Income Tax Records H8/H89		$ 18.00
885-1051	Payroll H8/H89		$ 50.00
885-1055	Inventory H8/H89	*	$ 30.00
885-1056	Mail List H8/H89	*	$ 30.00
885-1070	Disk XIV Home Finance H8/H89		$ 18.00
885-1071	SmBusPkg III 3 Disks H8/H19 or H89	*	$ 75.00
885-1091	Grade and Score Keeping	*	$ 30.00
885-1097	Educational Quiz Disk H89 or H8/H19	*	$ 20.00

DATA BASE MANAGEMENT SYSTEMS (DBMS)

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

AMATEUR RADIO

885-1023	RTTY Disk H8 Only	$ 22.00
883-1106	Morse-89 H8/H19 or H89	$ 20.00

* Means MBASIC is required

H11 SOFTWARE

885-1008	Volume I Documentation and Program Listings (some for H11)	$ 9.00
885-1033	HT-11 Disk I	$ 19.00
885-1053	H11/H19 Support Package (Modem, DCOPY, SCREEN, USRLIB, others)	$ 20.00

CP/M SOFTWARE (5-inch only)

885-1201	CP/M (TM) Volumes H1 and H2	%	$ 21.00
885-1202	CP/M Volumes 4 and 21-C	%%	$ 21.00
885-1203	CP/M Volumes 21-A and B	%%	$ 21.00
885-1204	CP/M Volumes 26/27-A and B	%%	$ 21.00
885-1205	CP/M Volumes 26/27-C and D	%%	$ 21.00
885-1206	CP/M Games Disk	%%	$ 21.00

The above CP/M products are 2 disks each.

885-1207	TERM and H8COPY	$ 20.00
885-1208	HUG Fig-Forth H8/H89 2 Disks	$ 40.00
885-1209	Dungeons and Dragons Game MBASIC and H89 or H8/H19	$ 20.00
885-1210	HUG Editor	$ 20.00
885-1211	Sea Battle Game for CP/M	$ 20.00
885-1212	CP/M Utilities I	$ 20.00
885-1213	CP/M Disk Utilities	$ 20.00
885-1214	Amateur Radio Logbook	$ 30.00
885-1215	BASIC-E	$ 20.00
885-1216	HUG CP/M BIOS for CP/M 2.2.03 (2 disks)	$ 40.00
885-1217	HUG Disk Duplication Utilities	$ 20.00

% Means CP/M 1.43 only (ORG-4200)
%% Means CP/M 1.43 or 2.2 (Heath)
Other CP/M disks are for 2.2

MISCELLANEOUS

885-0017	H8 Poster	$ 2.95
885-0018	H89 Poster	$ 2.95
885-0019	Color Graphics Poster	$ 2.95
885-4	HUG Binder	$ 5.75
885-4001	REMark VOLUME I	$ 20.00
885-4002	REMark VOLUME II	$ 20.00

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

N A T I O N A L H U G C O N F E R E N C E
"How Much Are You Interested?" Card

Name _____

Address _____

City _____ State ____ Zip _____

|__| Yes! I'll attend the **First Annual National HUG Conference** and here's my 15 bucks. I know I can eat that much.

|__| Can't make up my mind that far in advance.

|__| No chance! But tell me about it in REMark.

|__| Send me Hotel reservation information.

|__| I'll Sleep in the car (or elsewhere).

|__| I'll stay over Monday and take the tour of Heath.

Fold, staple, mutilate and send to us right away! All payments must be by check or money order. Send to National HUG Conference, Heath Users' Group, Hilltop Road, St. Joseph, MI 49085.

BULK RATE
U.S. Postage
PAID
Heath Users' Group

Heath
Users'
Group
Hilltop Road
St. Joseph MI 49085

POSTMASTER: If undeliverable,
please do not return.

885-2026