

## PORT TO PORTAL —Editorial

You'll find that this issue's contents quite distinguish it from the last. Still, just two issues can't really give a good indication of the range of material to come. Let the unfolding of future issues disclose that. I can at least say, for the reassurance of novices out there, that novices are on my mind at all times. Even an "advanced" article shouldn't be so abstruse that a beginner can't grasp **something** of value from it. For example, some who have never patched a program may consider patch procedures an advanced area. But even a novice should be able to perform the patch to QUERY in this issue if he follows the instructions. The Q/A column, meanwhile, is always a good place for novices to browse. Its repertory is mixed, because experts must ask questions too. All tyros need are big ears.

Some of you have asked about contributing articles for publication. Yes, if you have something you think would interest the readership, please send it. Maybe someone else can benefit from a problem you solved, or can be spared reinventing the wheel. Depending on originality and length we'll try to work it in. The ideal length for an article is a function of its "payload" and the space available in a particular issue. But that's mostly **my** problem — if you want to send something, just send it; we can discuss details later if necessary. Don't let length stop you — either way.

— Hank Lotz

## THE EIGHT-BIT R/W —Letters

[Send letters to the address on the last page. Letters are printed in full or in part. If you don't want a letter published, mark it "Not for Publication". For a personal reply you must enclose a self-addressed, stamped envelope. —Ed.]

## Public-Domain FORTH

[From David Fisher / 2300 SW 43rd Avenue / Ft. Lauderdale, FL 33317]... I am looking for the command set of FORTH's editor and operating system. If anyone can help it would be greatly appreciated. I have Lee Brodie's book, Starting FORTH, but the commands he has listed don't work on my public-domain FORTH system.

## Trionyx Outreach

[From Jared Freeman / 600 Ginger Trail / Lake Zurich, IL 60047]... My H8 is relatively new to me and I am still trying to get it to run properly with the Trionyx boards. The people at Trionyx have been very helpful over the phone, but California is a long way away. [If you write to Jared, it could be a potential start for all those with Trionyx questions to get in contact. —Ed.]

## Spikes Cause Garbage

[From Phil McCrum / 35 Rogers Court / Brea, CA 92621]... Does anyone have a SIGMASOFT and SYSTEMS

Interactive Graphic/Pseudo Disk/Spooler board installed on either their H8 (preferably) or H89? I would like to be in touch with anyone as to any problems they may be having with their subsystem and any successes they may have had. I have a problem with spikes which cause garbage to appear through the IG circuitry whether I am using the IG or not at the time; it is very annoying: any suggestions? Also, I am planning on getting the SIGMASOFT and SYSTEMS Hard Drive Subsystem. Does anyone out there have one? Are there any reasons I shouldn't get one?

## Roadmap Needed

[From Lois B. Laulicht / Conley Run Road / Valley Head, WV 26294]... As a confirmed "89'er" albeit an ignorant one, I have a couple of pressing needs and/or questions. My computer, a Heathkit, was purchased used a few years back. Quite recently I purchased a pair of Shugart #455 disk drives in order to upgrade storage capacity by using Livingston Lab's BIOS80. Notwithstanding a few hardware problems like the power supply reversed for one drive, we, with the help of a super technician at H/Z got them up and running.

Well, sort of! I now have 270K on all three drives because I cannot get the BIOS80 to boot. I've used every variant of MOVCPM17 in both the H/Z manual, and Cortesi and Zacks. I need a roadmap. More importantly I need to understand how to use DDT so that I don't need MOVCPM17 at all. Can anyone recommend a DDT tutorial for nonprogramming types like myself? The documentation on DDT is really very bad... Dr. Wm. Parke's Exploring With DDT reprinted in CHUG Dec. 1983 helps a lot but assumes more knowledge than I have. Help!!

Back to the problem of MOVCPM. Why is it not picking up the proper system info and writing it to disk? [I talked to Lois, and I think she's aware that MOVCPM **itself** does not do the writing to disk. —Ed.] The operating system that is running is quite changed from the distribution BIOS.SYS but BIOS80 was assembled with the MAKEBIOS.COM.

## THE 8-BIT IOWAN

Kirk L Thompson  
#6 West Branch Mob Hom Vil  
West Branch, IA 52358

Hi from overcast Arkansas (where I'm writing this),

Thanks to you who wrote and expressed your appreciations and corrections. The feedback gives me some idea of how I'm serving your needs. I'll try to include some of these next time.

**Soft-Sector (CP/M).** But last time, I started discussing setting up soft-sector drives. Here, I'll restrict myself to the most widely used CP/M, plain-jane Zenith's, from hard-sector. My objective is to configure a **bootable**, soft-sector system. I'm presuming you've installed a soft-sector controller board, either Heath's old Z89-37 or the replacement

from TMSI, and have at least one drive connected to it. Even an old Siemens, if you have a spare "hard-sector" drive handy, will do! I'm also presuming you've installed the MTR-90 chips that came with the soft-sector controller. These chips contain a command you'll need during this process.

But I have one more item for you to check in the hardware before we proceed. Which controller board have you set as your primary system? This is determined at section 4 of dip switch SW-501 on the H-89's CPU board. If open, you boot from hard-sector; if closed, from soft. It doesn't make any difference **which** you boot from, so long as you know which is **primary**. This is because you can use the "BS" (boot secondary device) command at the "H:" prompt to boot the other system.

If you have an H8, don't despair! The software procedure should work as I describe, only the hardware will be different.

**Implementing System Software.** If you have version 2.2.04, you should try the "SETUP" disk. I did, but it wouldn't boot because of hard errors, so I can't say if it will do the job. (In fact, I would like to "rent" this disk to see if it will; write me to make arrangements — no SASE is necessary!) The procedure, below, will work with 2.2.03 but not 2.2.04; the BIOS.ASM file of the latter is 6K larger than the former, which makes it too large to reconfigure on standard hard-sector drives.

For this procedure, lay out four (4) disks as follows:

- A. a bootable disk containing these .COM files: ASM, CONFIGUR, FORMAT, MOVCPM37, PIP, STAT, SUBMIT, and SYSGEN;
- B. a formatted disk with BIOS.ASM, MAKEBIOS.COM, MAKEBIOS.SUB, and PREL.COM (a dup of Distribution Disk III);
- C. an empty, formatted disk for the new BIOS; and
- D. a double-density-quality disk for your new system, to be formatted later.

None of these should be write-protected.

**1. MAKEBIOS.** Everyone complains about this program, but it's also the only way to configure BIOS.SYS to recognize drive types other than those the distribution disks came on. Boot disk A and put disk B in your second drive. I strongly recommend using a minimum of two drives for this operation. If you don't, you'll be doing a slug of disk swaps as it runs; even with two drives I had 12! You could even connect that "soft-sector" drive to the hard-sector board as a temporary stand-in; just be sure to configure it if you do.

It might also be worthwhile, when CONFIGUR signs on, to go in and note your present configuration on a slip of paper. You'll need to reenter this information into the new BIOS later. To start the MAKEBIOS procedure, type in the command line:

SUBMIT B:MAKEBIOS C: B:

SUBMIT starts processing a batch file; swap disks as instructed.

The procedure will present you with a menu. Select option "E" to include **both** hard- and soft-sector disks in your system. Except for disk swaps, the operation will now run on its own. This takes about 12 minutes, so be patient. The rewards are

worth it!

**2. STAT.** When MAKEBIOS is complete, you have to change the existing BIOS from a read-only into a read/write file so you can replace it with the new one. Type "STAT BIOS.SYS<RETURN>" and look at the listing. If "R/O" is just to the left of the file name, convert it to read/write with:

STAT BIOS.SYS \$R/W

Now you're ready to bring over the new BIOS you just created.

**3. PIP.** So put disk C in drive B:, warm-boot with ctrl-C, and use PIP to copy it from the scratch disk to the system disk by typing:

PIP A:=B:BIOS.SYS[RV]

Once this is complete, reset the system, set up your soft-sector drive if necessary, and reboot.

**4. CONFIGUR.** When you reboot, watch what CONFIGUR does when it checks the hardware. You should see a report appear on the screen of how many soft-sector drives it found. If this **doesn't** happen, the new BIOS doesn't recognize the soft-sector controller. There may be something wrong with the controller board installation, so check your installation documentation!

But if the report appears, you can reconfigure your system. So press "N" at CONFIGUR's prompt since you don't want the standard system. And go into options "A" and "C" to reset the terminal, printer, and I/O to what you had in your old BIOS. Now enter menu selection "B" and reconfigure your hard-sector drives to what they were. Before you exit with "Y", set your soft-sector drives for step rate and track density. For most drives, you can set a 6-ms step rate, but the Siemens should be left at no faster than, say, 20 ms. Be sure the track density is correct for the drives you have. Finally, save the new configuration to disk and memory.

Now you can check out the existence of your first soft-sector drive by typing "D:" at CP/M's prompt. The drive's light should come on and you will get a "Bdos Err on D: Select" message. Press <RETURN> to warm-boot. If you have more than one drive, repeat this with "E:" and "F:".

**5. FORMAT.** Now that the drives are installed, you're ready to prepare a bootable soft-sector disk. The first thing to do, of course, is format one, so call FORMAT, but use drive D: for the operation.

The next prompt you see is, "Which density? (S=single, D=double):". Actually, you have a third option, "extended double," which can be obtained by pressing "E". This density provides a bit more capacity, but at the cost of a larger sector size, so small files are stored less efficiently. I've run it without trouble on my 40-track drives for quite some time. So select either "D" or "E". You also are prompted for the number of sides; this should be the actual number your drive is able to write.

**6. MOVCPM37.** You're ready to create a bootable operating system in memory, configured for maximum RAM and soft-sector disk. All you need do is key in the command line:

MOVCPM37 \*

The asterisk allocates all of memory for the system.

**7. SYSGEN.** The next thing to perform is moving the operating system in memory to disk. But you **must** do this immediately after running MOVCPM37. If you do **anything** else, the new BIOS will be

overwritten! So call SYSGEN and press <RETURN> at the first prompt. This just tells the program that you're getting the operating system from memory. At the next prompt, enter the letter D. The program now transfers the kernel to that disk.

But one thing SYSGEN doesn't copy over is the new BIOS, so you must do this manually using PIP by typing:

```
PIP D:=BIOS.SYS[RV]
```

and there is one final thing yet to do.

**8. Reboot.** The proof is in the pudding! Reset the system. If you haven't touched SW501, enter "BS" and <RETURN>. If you set SW501 to boot from the new controller, boot normally. Your new, soft-sector system should now come up as it does from hard-sector drives.

So there you have it! But before going any further, you should change the read/write attribute of your new BIOS.SYS by typing:

```
D:STAT BIOS.SYS $R/O
```

and use PIP to copy all of the files from your old system disks to the new. And now you can FORMAT and SYSGEN soft-sector disks, as well as move your applications to your new system. It's easy, too, since you have direct access to your hard-sector disks in the old drives.

But if you have problems with this procedure, write; just be sure to enclose an SASE!

**In the Queue.** I'll shortly take you through a similar procedure for HDOS and show you why the "HOS-5-UP" disk from Heath is no loss if you didn't get it!

So see you next time!

Kirk

#### 4MHz-SPEEDUP INDEX

It seems most H-89 owners have either converted to 4MHz, or are interested in learning all they can about it. The following may help. It's a list of references to 4MHz in REMark issues. Those marked "Letter" are letters to BUGGIN' HUG. Most of the others should be articles.

REMark No.	Page
Nov '82	25 (Mod article by P. Swayne)
Dec '82	35 (Correction to Nov '82)
Feb '83	51 (Letter)
Mar '83	21 (Update to Nov '82 article)
Jul '83	45
	92 (Letter)
	95 (Letter)
Oct '83	91 (Letter on Nov '82 article)
	96 (Related to Mar '83 mod article)
Mar '84	61 (Letter)
Apr '84	20
Feb '85	6 (Letter -- PASCAL)
Mar '85	59
Oct '86	60

If you look these up in earnest, better look at all of them. Some references contain important corrections to others, and they may not all be annotated here. —HL.

#### MISCELLANY

**Still available?** Kirk Thompson stumbled across an address of a place that sells WORDSTAR. He didn't say for which system, but they'd be worth writing to. This was in the 3 March 1986 Infoworld, so we can only hope it's current at this late date. It is: Broadreach / 534 Industry Drive / Seattle, WA 98188. They also allegedly sell MAILMERGE, CORRECTSTAR, and an update to WORDSTAR ver. 3.31.

**Diskettes for sale, hard-sectored variety:** Last time, I mentioned the excellent SENTINEL hard-sectored diskettes Quikdata sells. Well, Ronald Pannatoni sent in 2 more sources of 10-hole hard-sectored diskettes. Both have \$25 min. orders & charge for shipping upon delivery (UPS). Per-disk prices are listed, but disks sell in boxes of ten only.

Global	Qty:	10	20-50	60-90	100 up
-----	Prices:	\$1.49	1.39	1.29	1.19
Order Nos:		C1925V = Verbatim;		C1925M = Maxell	

UARCO	Qty:	10	20-40	50-90	100 up
-----	Prices:	\$2.80	2.65	2.50	2.30
Order No:		2540 = Verbatim Datalife			

Don't know what differences, if any, there are in the two "Verbatims". And I've never tried any of the above diskettes myself. Addresses:

Global Computer Supplies / 45 S. Service Rd. / Dept 62 / Plainview, NY 11803 / 1-800-8-GLOBAL;  
UARCO Computer Supplies / 121 North Ninth St. / DeKalb, IL 60115 / 1-800-435-0713

**Adventures in Heathland (and elsewhere):** This may answer a number of questions -- questions about new printers, Heath repair policy, Heath's legal obligation, etc. One reader asks if Heath will still repair individual boards (as from H-8's or H-89's) without having to ship the whole machine back. I had a chat with the manager of the local Heath Store (well I still call it that; rolls more easily off the tongue). Benton Harbor will repair individual boards, but I'd write them first. Some local H/Z Stores may do the boards too, if they still have test fixtures. Whether or not a store has the required test facilities depends on the individual manager of that store. The store can also look at the board by opportunistically using another customer's machine, if one is in for repair at the time. What about servicing a machine modified by the user? The factory (Benton Harbor) won't touch it. You have to put everything back to standard before shipping it back! Some local stores will fix user-modified units. The stores will fix 8-bit machines with everything but a hard disk drive. (Unquote.) Ok, now, parts-availability? "Only active parts are stocked." (Whatever that means.) I assume the cabinet isn't an "active" part, that makes some kind of sense. (But watch them probably stock it!) So I asked about CRT's and got, Yes that's an active part. Asked if Heath had a legal obligation to repair its machines for seven years after stopping production (e.g. H-8's, H-89's) my man replied in the negative, claiming it was merely Heath's policy

to support them for a period. I guess no matter whom you ask from Heath you'll get the same answer. Perhaps an attorney could give an objective answer; I know our readership includes same.

A reader from Michigan wants to buy a letter-quality printer cheap, to just plug into an H-89A port and go. Not an unreasonable request, but this **implies to me** a serial printer, and aren't most daisywheel printers on the market now parallel, or expensive, or both? Well my local Heath store now (Dec 1986) sells an Epson DX-10 **daisywheel** printer (Diablo 630-compatible) for only \$249, but it's not serial. Comes **with tractor** and standard parallel **cable** though! All you need now is an easy parallel conversion like Quick-P from McGaffey Engineering (\$69.50). Write McGaffey at 1556 Halford Ave., Suite 169, Santa Clara, CA 95051. Or you can spend \$499 for the great new Brother HR-20, a really nice daisywheel job that comes **standard** with **both** serial and parallel ports, according to my Computerland rep. He says the HR-20 replaces the **now-discontinued** HR-15 and uses the same print-wheels, ribbons, and plugs. If you do get a **serial** printer, Kirk Thompson reminds us that the RS-232 standard allows for variations in handshaking configurations, and he recommends as an aid, Martin Seyer's **RS-232 Made Easy: Connecting Computers, Printers, Terminals, and Modems** (Prentice-Hall, 1984) which contains tables and charts for interfacing many peripherals to various micros, including the H8 and H89.

**Your Move:** Here's one for all you with chess programs, or knowledge of same. The checkmate with King, Knight, and Bishop to King is **definitely** forceable, but it's a long, intricate process. A superior chess player may gain that situation by the end game, **yet only draws** if he cannot execute this checkmate! Thus knowing it is **mandatory**; but few chess fans, in my experience, can do it. (I've worked at it, but haven't perfected it yet.) Can any of the chess software you have, or any self-contained electronic game on the market you know of, deliver that mate?

**HDOS users:** About the CP/M MBASIC tip from the last issue (page 7, how to change a single line number in BASIC). Kirk Thompson says it **does** also work under HDOS MBASIC.

**Answer to Cursor Position Question:** Issue #1's Q/A column wanted to know how to use the H8/H19 or H89 Cursor Position Report without hitting RETURN. Adolph Stumpf came to the rescue (and then some) with the following info. He writes:

"A Cursor Position Report can be obtained by a program without an echo to the screen. This routine will do it in MBASIC by using the INPUT\$ function instead of LINE INPUT:

```
100 PRINT CHR$(27)+"n"
200 A$=INPUT$(4)
```

The cursor position can then be easily extracted by using the MID\$ function.

```
300 FOR I=1 TO 4
```

```
400 PRINT ASC(MID$(A$,I,1)),
500 NEXT I
```

When the program is executed with the cursor in the lower left-hand corner of the screen the display is:

```
27 89 55 32
```

which is equivalent to ESC Y line #24 column #1.

"In [HDOS] assembly language the following routine should work:

SCALL	.SCIN	GET CHAR FROM TERMINAL
JC	*-2	
CPI	ESC	IS IT AN ESC?
JNZ	OUT	EXIT ROUTINE IF NOT
SCALL	.SCIN	GET SECOND CHARACTER
JC	*-2	
CPI	'Y'	IS IT A Y?
JNZ	OUT	EXIT IF NOT
SCALL	.SCIN	GET CURSOR LINE POSITION
JC	*-2	
STA	LINE	STORE LINE NUMBER
SCALL	.SCIN	GET CURSOR COL. POSITION
JC	*-2	
STA	COL	STORE COLUMN NUMBER "

Adolph says he hasn't verified this assembly routine, but has used a similar routine to read terminal escape sequences. I doctored up the BASIC example a bit to try to demonstrate that the cursor can be **anywhere**. After typing RUN is still the **only** place you have to hit RETURN. Also, I added comments:

```
100 R=11 : C=40 'PROGRAM SETS (OR READS) ROW AND COL
200 R$=CHR$(31+R) : C$=CHR$(31+C) 'FORM REQD STRINGS
300 PRINT CHR$(27)+"Y"+R$+C$; 'PUT CURSOR THERE
400 PRINT CHR$(27)+"n" 'ASK TERMINAL WHERE CURSOR IS
500 A$=INPUT$(4) 'THIS PICKS UP TERMINAL'S ANSWER
600 FOR I=1 TO 4 'USE RESULT AS DESIRED, (PRINT,ETC)
700 PRINT ASC(MID$(A$,I,1)), 'PRINT IS ONLY FOR DEMO
800 NEXT I
```

For some practical applications you could ignore the 27 89, and extract only the row and column, the last two numbers returned (each minus 31). Line 300 assumes you know about Direct Cursor Addressing (page 5-7, or page 11-12, in the H-89A Op Manual). When KEYMAP is toggled on, this program doesn't work for me. The initial ESC doesn't come through. I **speculate** that since KEYMAP likes to look at ESCAPES, it's holding back on us and doesn't think we really meant this one.

#### A PATCH for QUERY13'S ADD.COM PROGRAM

by Hank Lotz

[Adapted from my Pgh-HUG article of Feb 1986]

QUERY13 is a system of database management programs from HOYLE & HOYLE. One of those programs is named ADD.COM. The patch described here is for ADD.COM Version 1:00, for CP/M. The patch works beautifully under CP/M 2.2.03 and 2.2.04. My QUERY13 is on hard-sectored disks, and I use an H-89, but should it not also work on an H-8 then? I can't be certain if QUERY13 ADD.COM Version 1:00 is the same as under QUERY12. (The title on the screen comes up as



"QUERY!3", so if **that** was updated, perhaps some other code as well?)

**What This Patch Does.** ADD is used to put data into your QUERY!3 database. It appends new records to the database starting just after the last existing record. For example, if there are already 10 records in your data file, ADD wants to write number 11. This sequential ADDing of records is just fine; QUERY!3's SORT can rectify the order later, and other modules can do random access. However, ADD has a habit of asking, after each and every record you enter, whether you want to enter another record. The programmer obviously wanted to show the user a visible way out. Good programming form, true, but when you sit down to type in a big bundle of data, having to type "Y" after every record can get mildly frustrating. The 13-byte patch given here suppresses the question, "Another Record to Add (Y or N)?", at those times. You'll still have to type "R" to Record the latest entry onto disk, but you can enter the next record without other interruptions. When you have no more data to enter, you will use ADD's ESC-key feature to "abort the sequence" when ADD is expecting a field input. After the ESC, the patch does **not** suppress the question, "Another Record to Add (Y or N)?", so you can reply "N" to exit. If you've used the original, you should find the patched version a joy, because you are not prodded merely to continue, and an "exit" option can be summoned by the ESC key!

Want to try the patch? First, **copy** ADD.COM; never patch a distribution disk. This procedure assumes you have PIPped the **copy** of ADD.COM and a copy of CP/M's DDT.COM to a "scratch" disk, and have put that disk in drive B:, also selecting B: as your default drive. Use DDT as follows to make the changes. Always hit RETURN after each entry.

Computer types	You type
B>	DDT ADD.COM
DDT VERS 2.2	
NEXT PC	
1A00 0100	
-	S0385
0385 70	BE
0386 10	19
0387 FE	. (period)
-	S0570
0570 21	C3
0571 7D	95
0572 1A	10
0573 CD	. (period)
-	S0EBC
0EBC 09	0D
0EBD 09	80
0EBE 2D	21
0EBF 2D	7D
0EC0 2D	1A
0EC1 2D	C3
0EC2 2D	73
0EC3 2D	10
0EC4 2D	. (period)
-	<CONTROL-C> (to warm boot)
B>	SAVE 25 PADD.COM (Note use of new name for PATCHED copy.)

This completes the patch! PADD is now your new ADD. Note: In working this out I needed space to insert two opcodes. I isolated an area containing a row of dashes that underscored the second of two\* version-number references. The only consequence of this is that now the pretty dashed line won't show on the screen beneath that version-number message. (\* The version number appears only once if run from the QUERY menu.)

### BEEFING UP THE H89 IN THREE STEPS To Provide More Current, Reduce Heat, and Remove Fire Hazards by Dan Jerome

This new wiring procedure (authorized by the local Heathkit store) will insure that you do not have any dangerous, damaging fires in your computer. Installing the new bridge with greater current capacity will provide you with a current supply that can handle all your needs; past, present, and future. It will allow you to install just about any new hardware modules you desire without creating a shortage of current.

When Heathkit designed the H89 Computer prototype, about 1976 or so, it was the first and best "all-in-one" home computer on the market. The engineers calculated probable current supply figuring only on using the original parts, including: the CPU Card, the TLB Card, the triple output card, the hard-sector card, and one drive inside the computer case. In practice, their calculations proved out to be marginal for even that. The reason that fires start on an H89 is that the rear power supply is now getting more current through it than it was originally designed for. Instead of forming a complete and full contact, the parts used only form a point contact on a heavy-current line. After a few years, corrosion sets in and increases the resistance. Greater resistance causes heat. That kind of heat can start a fire. And that is why you should immediately make this modification if not sooner. You will always be glad you did. [Editor's note: I asked Dan whether an H89A would have design improvements over the older H89 in the power supply. He replied: "When the H89A comes out of the store it is **not** beefed up. The main thing that makes it an H89A is the special pains Heath went through to make it less of a radiation problem." He goes on to say that in **that** area (the EMI or RFI), "Our H89A's are still considered to be a very 'dirty' machine..."]

Anyone who built his own computer shouldn't have any problem putting in this mod.

**(STEP 1) Improving the Bridge Rectifier Circuit:** First, secure a new heavy-duty bridge rectifier from Radio Shack. Page 118 of their catalog dated October 1986 lists a 25-amp, 50 PIV (peak inverse voltage) full-wave bridge rectifier, part no. 276-1185, price: \$2.69. Substitute this unit for your standard-issue bridge rectifier, BR-1. There will be no electrical problems in making the substitution.

Installing it will require you to unplug the front cable assemblies, P102 and P104, and the rear

cable assemblies, P101 and P103. Then remove the power supply circuit card by taking off the four #6 screws, and swing the card to the side. This will give you access to the power supply heat sink, the large heavy-duty aluminum L-shaped bracket which holds the three voltage regulators and the bridge rectifier, and supports the power supply. After locating it, swing it away from the computer so that you can drill one hole in the appropriate position in order to punch a screw, lockwasher, and hex nut thru to attach the new BR-1. Use plenty of heat sink compound between the bridge rectifier and the power supply heat sink. If you don't have any heat sink compound, Radio Shack does. See their catalog, page 119 for part no. 276-1372, cost: \$1.39. (A mere pittance!)

Prior to rewiring the new bridge rectifier, you should also replace voltage regulator U101, located just above the bridge rectifier, with a different part number that can deliver more current. Also, all three voltage regulators should be equipped with large heat sinks. This will aid further in dissipating heat.

#### **(STEP 2) Improving the +5-volt Current Supply:**

The second step will also supply more current to your computer circuit card assemblies. It will also reduce the amount of heat that is generated by using your computer. This step consists of purchasing a new 5-volt voltage regulator to replace reference designation U101. This unit is positioned on the power supply heat sink assembly and is mounted just above the bridge rectifier, BR-1. One vendor with whom I have done business satisfactorily in the past and who can supply the needed part is: JDR Microdevices / 1224 S. Bascom Ave. / San Jose, CA 95128 / Phone: 1-800-538-5000. Part number is: 78H05K, regulator, voltage, 5-volts, 5-amps, cost: \$7.95. (Heathkit wanted \$18 for the same part when I checked last.)

You also probably want to order three heat sinks for each of the voltage regulators mounted on that power supply heat sink assembly unless you already have them installed. To secure these parts, first check with your local Heathkit store to see if you can obtain them there. Ask for part no. 215-658. If Heathkit doesn't have them, then try JDR or any other vendor on the phone. The kind of heat sink you want is a large one for TO-3 type devices.

**(STEP 3) Reassembly and Rewiring to Prevent Future Fire Hazards:** First reassemble the power supply heat sink to the chassis, using the four #6 threaded spacers, then put the power supply circuit card back in place, and connect the two cables at the front of the power supply.

When you rewire the new unit, bring all wires up from the chassis directly, omitting the use of the rear plug. The red wire must be soldered to the positive terminal on the bridge rectifier, and the black wire must be soldered to the negative terminal on the bridge. The yellow wires are "don't care". They could go in either way but they must also be soldered to the two remaining lugs on the new bridge rectifier.

**CAUTION:** Whenever soldering semiconductor devices,

use heat sinks to prevent the devices from becoming too hot and damaging them.

#### **NOW YOU CAN USE THE FUNCTION KEYS IN YOUR OWN PROGRAMMING!**

by Hank Lotz

In the past I've seen articles about the H8/H19/H89 function keys, giving methods that didn't work so well. Follow-up articles explained why they didn't work, then gave other methods that didn't work (or required some large, special resident utility). One bugaboo was that the 3-character-long alternate-keypad functions didn't work, especially with a BASIC interpreter. The characters came too fast, and often one was dropped. Well, I have two ideas that **do** work! One, I admit, is a purchased package. But I also want to show CP/M users how to define those same function keys in BASIC or FORTRAN programs they write themselves, by making use of an 8080 assembly language subroutine. HDOS programmers may get a very valid idea by extension. My subroutine for reading the function keys (including the 3-byte alternate-keypad keys) in **your own programs**, comes later in this article, so don't be scared away if this resembles a software review at first!

**The "purchased package":** For \$20 you can buy a **great** program that lets you use the H8/H19 or H89 function keys for everyday use. It's called KEYMAP, and I highly recommend it to you. KEYMAP (or even a similar utility) is an **absolute must** for H89/H19 users! CP/M KEYMAP is HUG Part No. 885-1230[-37]. No HDOS version exists that I know of. Other KEYMAP versions are available for the Z-100 and Z-150; see REMark, April 1985, pp 34-35. If you don't have and use it, you are wasting your time, causing yourself unnecessary frustration, and missing the beauty of function-key use for everyday work. To invest in a RAM-drive and still sit there typing "A:STAT \*.\*" all day, is **not** consistent behavior. (It's like owning an automatic garage door opener and having to get out of your car and walk over to push the button. I may sound lazy, but if you have it you might as well "stay in your car" when you use it.) A:STAT \*.\* is **fourteen keystrokes**, counting the SHIFT keys needed, and the CR at the end. If you had only **one** of HUG's utilities, it should be KEYMAP! Buy it, read the doc, and use the program! KEYMAP lets you define a function key (including the keypad keys) for up to ten characters. Are ten characters sometimes not enough for you? Just put the rest of the character string on an adjacent function key, and press the two keys in **sequence** for longer instructions. You can end a string with a CR, or not, as you wish. Not only that, you can **doubly define** each function key by designating one of the function keys as a "shift" which momentarily throws all function keys into their second defined meanings. This effectively doubles the number of function keys on your terminal! While you can program the keys any way you like, there are some ready-to-use configurations supplied. For example, a set for BASIC programming. The examples are legion, but think for a moment how nice it would be to type the ubiquitous CHR\$(, or GOSUB, with only one

keystroke. You may set the keys to be control characters also. I set one for CONTROL-X which gets a lot of use! Well, this was not meant to be a review; it was designed to whet your appetite for a terrific utility you really should have. So now let's move on to our own programming methods.

**Function Keys in Your Own Programs.** If you want to concoct your own program and use the function keys from within that program, I have an assembly language subroutine that will read the keys. The bytes read are passed back to the MAIN (or calling) program. You can then inspect the input with your calling program, test to see which key was hit, and act upon that any way you like. Again, I think some of the past difficulties with using the function keys with BASIC were related to the **3-byte alternate-keypad function keys**. Whether the hardware wasn't fast enough to catch all three bytes or whether the BASIC software was at fault is moot, because the present method works. But please understand that the code I give here is only one variation. For example, the way it branches when two consecutive ESC's are encountered can be easily changed if you want to customize program behavior. Also, since the subroutine (as written) rejects any character but ESC at the beginning of a string, only two parameters would really need to be passed in the CALL list; we could just **verify the presence** of that first ESC, in the subroutine. However, I'm leaving it this way to illustrate how to pass three parameters when using assembly language, because **actual examples** like this are **mighty scarce**. Actually, you can even get away with passing only one parameter — let the **subroutine** test which key was struck and have it return your own 1-byte identifier code. I do the key test in the **calling program** because I can write high-level code a lot faster. Who can't. I recognize that for many purposes we would **also** want the subroutine to be able to read **any** keyboard key, not just function keys, and indeed, the subroutine can be adapted for that feature. Presently, however, it is mainly to convey fundamental principles.

With compiled BASIC or FORTRAN, linking the compiled relocatable subroutine using L80 is straightforward. For a BASIC interpreter, however, it will be up to you to figure out how, and where, to load the assembly routine with the BASIC program. Some helpful discussions of such loading can be found in REMark (e.g., May 1985, p72; and Oct 1985, p21). The MICROSOFT BASIC-80 manual also discusses it (at least mentions it briefly). I did not test my subroutine with **interpreted** BASIC, but I feel if it doesn't work nothing will; and I see no reason why it wouldn't.

When the subroutine is called, only the 16-bit **address** of each variable in the CALL list is passed. This is the address of a single byte. If a variable occupies more than 1 byte, its first-byte (lowest memory) address is passed. Note that when the subroutine zeros out an argument, or otherwise assigns it a value, **only one byte** per argument is affected. For FKEYS to work, the passed variables (I, J, and K) can be either INTEGER\*1 or INTEGER\*2 in

FORTRAN. In BASIC they (I%,J%,K%) will be the equivalent of "INTEGER\*2", that is, 2-byte variables. INTEGER\*1 is safer because **only the low byte** of an INTEGER\*2 variable will get modified by the subroutine. For example, if "I" is INTEGER\*2, and you set it to 256 before the subroutine call, its value upon return will be 256 + 27 = 283. So if you're using FORTRAN, either set the arguments equal to zero before the call, or define them as INTEGER\*1. I guess in BASIC you're stuck with 2-byte variables, unless you can play some kind of game with them. They **will work** as 2 bytes, but I'd set them to zero early in the program in case the high byte contains garbage.

```
;SUBROUTINE FKEYS (Reads Function-Key Input).
; Version 1.00      8-21-86   Hank Lotz
; FKEYS zeros out all 3 parameters passed to it.
; FORTRAN Calling Sequence:  CALL FKEYS(I,J,K)
; MBASIC Calling Sequence:  CALL FKEYS(I%,J%,K%)
; Upon entering this subroutine, HL will contain
;   addr of 1st param, DE addr of 2nd, BC addr of 3rd
; EXAMPLES of VALUES RETURNED (I,J,K) :
; Example 1: Alt. kypd "ENTER": I=ESC, J="?", K="M"
; Example 2: Functn key f1:    I=ESC, J="S", K=0
;   (Unused parameters will be returned as zeros.)
;
ESC      EQU      27 ;ASCII decimal value
WBOOT    EQU      0 ;CP/M warm boot address
DIRIO     EQU      6 ;Direct Console I/O function
BDOS      EQU      5 ;BDOS address
;
; ENTRY FKEYS ;Keep label colons for M80.
FKEYS:   SHLD     ARG1 ;Store HL's value
        LXI      H,0 ;Initialize HL
        DAD      SP ;Put stack pointer into HL
        LXI      SP,STACK ;Point to our local stack
        PUSH     H ;Stash CP/M stack pointer
        PUSH     PSW ; and A-flag reg.
        XCHG     ;Swap HL with DE
        SHLD     ARG2 ;DE's value gets stored
        MOV      H,B
        MOV      L,C
        SHLD     ARG3 ;BC gets stored
;
;*** Zero out the 3 passed arguments
INIT:    LHLD     ARG1 ;Point to 1st param address
        MVI      M,0 ;Initialize 1st param
        LHLD     ARG2 ;Same for 2nd and 3rd params
        MVI      M,0
        LHLD     ARG3
        MVI      M,0
        JMP      FIRST ;Now to read a key
;
INPUT:    MVI      E,OFFH
        MVI      C,DIRIO
        CALL     BDOS
        ORA      A ;If 0 no char is ready
        JZ       INPUT
        RET      ;Now we have a char
;
FIRST:    CALL     INPUT
        CPI      3 ;Was it CTRL-C?
        JZ       SYSTEM ;Warm boot if CTRL-C entered
        CPI      ESC ;An ESC is imperative here
        JNZ      FIRST ;Ignore anything else
        LHLD     ARG1 ;Point to addr of 1st param.
```

```

MOV     M,A      ;Put the ESC there
;
CALL    INPUT    ;Get second char
CPI     ESC      ;ESC NOT allowed here
JZ      INIT     ;We must start over
LHLD    ARG2     ;Wasn't ESC, so take it
MOV     M,A      ;Got it
CPI     '?'      ;Alt keypad?
JNZ     CLNP     ;If not, we're finished
;
CALL    INPUT    ;Last char was ? (Alt kypd)
CPI     ESC      ;Can't allow an ESC now
JZ      INIT     ;Start over
LHLD    ARG3     ;Point to addr of 3rd param.
MOV     M,A      ;Put char there, whatever it is
                ; These have to be looked at
                ; in MAIN anyway.
CLNP:   POP      PSW      ;(cleanup)
        POP      H
        SPHL     ;Set stack pointer to old HL
        RET      ;RET always POPs from stack
                ; to program counter(PC reg)
SYSTEM: POP      PSW
        POP      H
        SPHL
        JMP     WBOOT
;
ARG1:   DS       2        ;Will hold 1st param address
ARG2:   DS       2        ;Address of 2nd param.
ARG3:   DS       2        ;Address of 3rd param.
        DS       32       ;Allot stack space
STACK  EQU      $        ;"Bottom" of stack goes here
        END

```

Now, as a demo, here's a simple calling program and some instructions for compiling and loading it:

```

PROGRAM KEYTST
INTEGER*1 I,J,K
1      WRITE(1,100)
100    FORMAT(' PRESS A FUNCTION KEY:')
        CALL FKEYS(I,J,K)
        WRITE(1,200)I,J,K
C THIS SHOWS THE DECIMAL BYTES OF THE FUNCTION KEY.
C (K WILL BE ZERO FOR THE 2-BYTE KEYS.)
200    FORMAT(' THE SEQUENCE REC'D WAS: ',3I6/)
C TEST WHICH KEY WAS STRUCK:
      Use your customized IF(I.EQ. etc... (May be
      logic to branch on IF(J.EQ. etc... omitted for
      subroutine output. IF(K.EQ. etc... this demo.)
      GO TO 1
C HIT CONTROL-C TO EXIT.
      END

```

Here is but one scenario for didactic purposes:  
Name the FKEYS subroutine source "FKEYS.MAC" and put it on drive B:. KEYTST.FOR is the above, also on B:. The compiler, F80; the loader, L80; and the MACRO-80 assembler, M80, are on drive A:. The FORLIB.REL library is also on A:. If everything goes smoothly, you can do it exactly like this:

```

B>A:M80 =FKEYS (A space before the "=" is req'd)
B>A:F80 =KEYTST
B>A:L80 KEYTST/N,KEYTST,FKEYS,A:FORLIB/S/E
B>KEYTST (This runs KEYTST.COM just created)

```

## QUESTIONS and ANSWERS

**Q — What is an "overlay"?**

**A —** Any time "data" is overwritten by other data (in memory or on disk, and for any purpose) it can be said to be **overlaid**, and this applies equally to data per se, or program code. **Overlays** are frequently used when too little memory is available to accommodate a large program. The program is broken into segments or subroutines which are loaded from disk when called, **overlying** a previous subroutine in memory. If the previous ("destroyed") subroutine is ever needed **again**, it must be reread to overlay the current one. Also it is often a valid practice to overlay an array of **data** with other data read from disk, to avoid declaring a larger array in limited memory. Overlay techniques are **wonderful** for 64K computers. Devising overlays requires skill, insight, and caution, and there is a trade-off between saving memory and saving time.

**Q — I need a database program that does some math.**

**A —** The person who wrote this has QUERY13 from Hoyle & Hoyle and I'll just bet he doesn't have their extension package, CALC, which **does some math** on the fields and would work for business applications he described in his note. CALC also has string search features which the WRITER module does not have, and thus can print or process selected fields or records.

**Q — Is it true that CP/M can address only 64K?**

**A —** Yes, for 8-bit processors like Z80's and 8080's in our H-89's and H-8's. Their SP and PC registers are two bytes (16 bits) wide, and so can point to any address from 0 thru 65535. That's all you can do with 16 bits. Schemes exist for using **banks** of 64K, but only one 64K bank is selected at a time.

**Q — Who will perform repair service on disk drives?**

**A —** For this you can write Floppy Disk Services, 39 Everett Dr., Bldg D, Lawrenceville, NJ 08648

**Q — I would like to know why the only way I can get a bootable system from LLL BIOS80 is by going through the whole installation procedure. MOVCPM80 gives a system but not a bootable one.**

**A —** You must not neglect to SYSGEN a configured system **directly** after running MOVCPM. No SYSGEN, no bootable disk!

**THE STAUNCH 8/89'er** (Hank Lotz, Editor; 2024 Sampson Street; Pgh., PA 15221) is a quarterly newsletter about 8-bit H/Z computers. Price: \$5.00/year. Single copies: \$2.00. Neither this newsletter nor its editor is responsible for damages or losses resulting from use of any information presented herein. Information in this publication may be reprinted only if credit is given to THE STAUNCH 8/89'er, listing its address. Credit should also be given to authors and other sources of reprinted material, if known. CP/M is a registered trademark of Digital Research, Inc. REMark is a registered trademark of HUG (Heath/Zenith Users' Group). -30-