MACROTECH'S V-RAM

A DUAL-PURPOSE BOARD

Jay Vilhena

half a megabyte of Static RAM for your S-100 system will cost about $800 (usually at discount prices) and from 1 to 8 slots of your motherboard. You can, with little difficulty, save the extra 6 or 7 slots by buying high-density boards, but you will not be able to shave much off the $800. However, a way exists to get a free RAM-disk tossed in the deal. It's called V-RAM.

V-RAM FEATURES

As you might have noticed, high-density Static S-100 RAM boards have started to appear. Not too long ago, the maximum Static RAM that you could get in one S-100 slot was 64K. Now your slot can accommodate from ½ Meg to... hold on till the next issue, I've been asked not to tell yet.

Anyway, MACROTECH got ahead of the pack and, as early as last year, had a 512K V-RAM on the market. Now, while other first-version high-density Static RAM boards are only starting to show up, the V-RAM is already on its second, debugged version.

The V-RAM is unique and a deal. Its price does not differ significantly from other Static RAM boards of the same size, but the V-RAM carries the extra capability of doubling as a RAM-disk with on-board battery back-up. A simple operation, removing or installing a couple of jumpers, decides whether the board becomes RAM-disk, system RAM, or both.

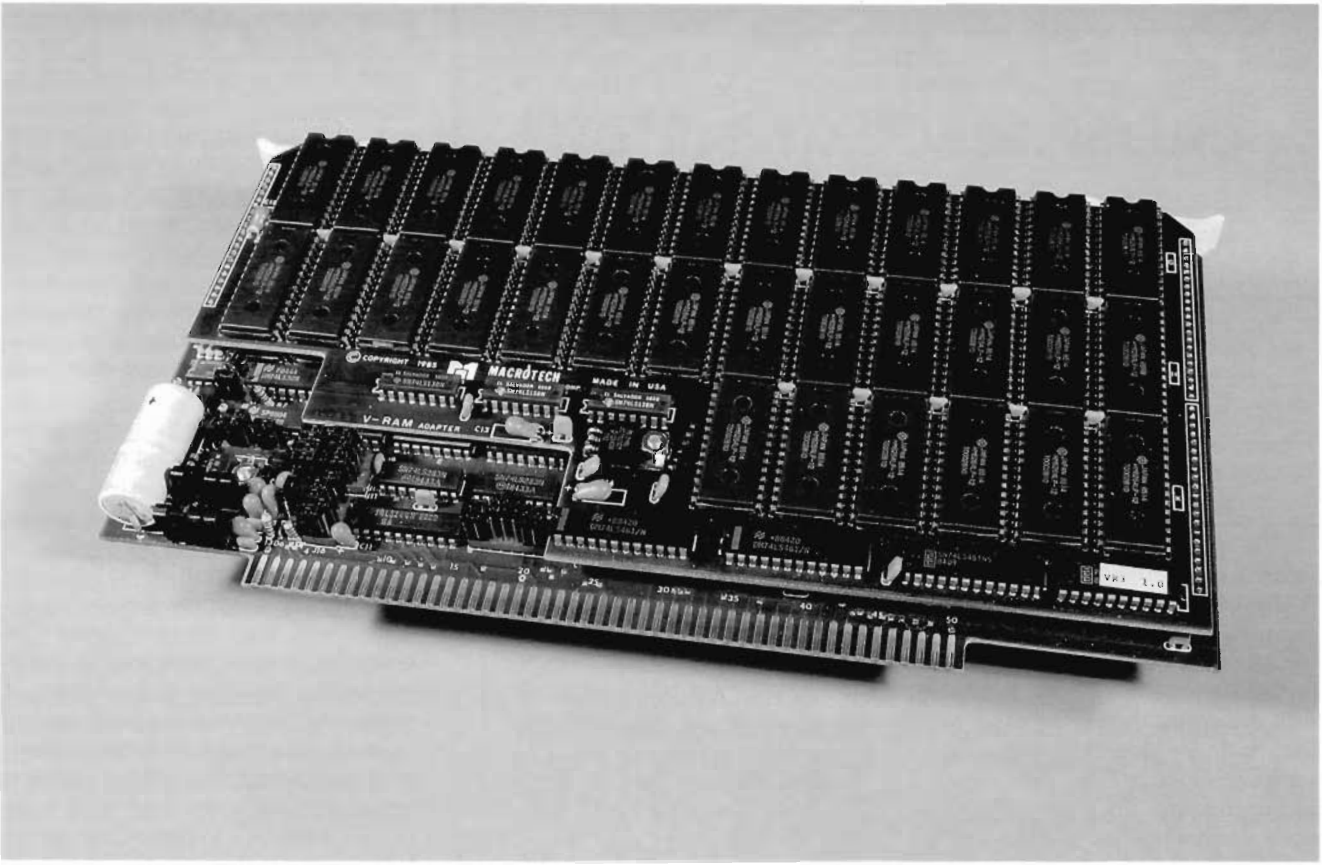
V-RAM as System Memory

When jumper J12 on the V-RAM is set to the right, the board performs as system memory. An extra ½ Megabyte of highly-reliable and fast Static RAM becomes available to your system. The V-RAM adheres to IEEE-696 specification so there should be no incompatibility problems adopting it as the system memory of any S-100 computer that follows the standard. The board can be accessed by either 8-bit or 16-bit CPUs.

For older S-100 systems, or new systems that do not fully comply with the 696 standard, the V-RAM manual describes a few common remedies to get the board to run properly. But, even if the remedy necessary for it to run on your system is not listed, I'm certain that Macrotech is willing to answer questions that you might have on how to adapt the board for your S-100 computer.

The V-RAM board can be set so that its address starts at any 64K boundary within the 16 Megabytes of extended addressing. Explanation: There are 24 lines in the S-100 bus that are used to select the address of each byte in memory. Since each line can represent a 1 or a 0, the 24 lines can represent any binary number from 000000000000000000000000 to 111111111111111111111111. That's a total of 16,777,216, or 2^{24}, possible addresses. Storing one byte at each address gives a potential memory capacity of approximately 16 Megabytes. Originally, the S-100 bus had only 16 address lines; when 24 address lines were defined, the capability to address all 24 was named extended addressing

So that multiple memory boards can be used in a computer system, each board must be assigned a starting address from the possible 16+ million. The V-RAM can be set up to start at any address that is a multiple of 65,536 (the same as 10000 in hexadecimal). Note that each block of 65,536 addresses can hold 65,536 bytes which is defined as 64K. This



is why it is said that the base address of the board can be set to any 64K boundary. For example, if you already had a 128K memory board in your system, the last address available would be address 131,071, so you would want to set the V-RAM to start at address 131,072 (20000 hexadecimal). This is the boundary where the third 64K (bank) starts. You could go on adding memory boards, with each base address starting where the last board ended, until you filled the entire 16-Megabyte memory space (you would need 32 V-RAM boards for example).

And by the way, the V-RAM is also available as a 256K memory board, but in this case there is no RAM-disk capability.

V-RAM as RAM-Disk

Set jumper J13 to the right, and your V-RAM becomes a RAM-disk or, as Macrotech calls it, a Macro-Drive. A RAM-disk acts exactly like a disk system except that, instead of a disk drive and diskettes, you have memory chips on a board, extremely fast access speed, and no moving parts.

The Macro-Drive feature of the V-RAM is designed to be compatible with any software written for CompuPro's M-Drive/H, a popular Dynamic-memory RAM-disk. Up to eight V-RAMs (or a combination of V-RAM and M-Drive/H boards) can be used together for a maximum of 4 megabytes of RAM-disk.

Like the M-Drive/H, the V-RAM Macro-Drive communicates with the master CPU through two S-100 I/O ports. This information is irrelevant if you have RAM-disk driver software (simply make certain that the port address jumpers are set correctly — see below). However, if you need to write your own drivers, examine the article and example listings by Leonard Schwab in this issue of S-100 Journal.

A RAM-disk is one of those items that, once you have one, you wonder forever how you were able to survive without it. The increase in execution speed is more dramatic, obviously, in operations that use the disk intensively, as in compiling programs. But even with trivial operations, such as loading a program or a database into memory, a RAM-disk can save enough time to more than justify its purchase.

To give a rough idea of just how much time a V-RAM Macro-Drive can save, I made two informal comparisons of operations involving disk access (V-RAM versus a QUME 8″ floppy drive). In the first comparison, I compiled and linked BILL, my C program that plucks from a subscriber file naughty S-100 Journal subscribers who have not paid for their subscriptions and sends them a bill. In the second comparison, I simply measured how long it took to load into memory a text file of about 100K. The results are in Table 1 and speak for themselves. Obviously these times are very dependent on what programs are used, the type of hardware, the operating system, and many other variables, so the absolute values are meaningless, but the relative values demonstrate the point.

RAM-Disk AND System Memory

By Setting both jumpers, J12 and J13, to the right, the V-RAM is ca-

	Using Qume 8-inch Drive	Using V-RAM Macro-Drive
Compile and Link BILL	4 min. 27 sec.	39 sec.
Load a Text File	20 sec.	12 sec.

Table 1. *The V-RAM versus a Qume 8-inch Drive. Comparison of the time required to compile and link a C program and to load a text file into memory.*

pable of performing simultaneously as RAM-disk and system memory. Unfortunately, while on this mode, there is no means of physically separating the system memory from the RAM-disk area, i.e., they overlap through the whole ½-Megabyte range. Thus, for many applications, this feature is inconsequential since memory operations easily mess up the data of the RAM-disk and vice versa. I suppose there might be some way of writing memory/RAM-disk management software to take better advantage of this feature, but it seems hardly worth the trouble.

Nevertheless, it should be very useful to leave J12 and J13 on (to the right) so that both the RAM-disk and the 512K of system memory are always accessible without having to open the computer to reset the jumpers. The key here, to avoid loss of data, is NOT to use the two features simultaneously. For example: you have 1 M-byte of memory and the upper 512K is a V-RAM board. Configure two versions of your operating system, one for 512K and the other for 1024K (1M) of memory. Keep (or install) the RAM-disk drivers with the 512K version. Now, when you want to use 512K of memory and the RAM-disk, boot with the 512K version of the operating system. And when you want to use 1024K of memory and no RAM-disk, boot with the 1024K version. You never have to bother with the jumpers.

In addition, the V-RAM capability of performing simultaneously as disk and memory, could be useful to hackers and in software development since it permits looking at the contents of a disk file while accessing the board as system memory.

Battery Back-Up

A great disadvantage of Ram-disks has been the inability to hold data after the computer goes off or even during a short power failure. You had to constantly be saving your RAM-disk files to real disks lest your files were no more. Some solutions to this problem have appeared, but usually they involve the extra purchase of a battery, connecting it to the board via a wire, finding a place to mount it on the frame, worrying about replacing it, and other hassles.

Macrotech solved the problem. A relevant feature of the V-RAM is its standard battery-support circuitry and Ni-Cd battery conveniently premounted on the board. The board circuitry takes care of automatically recharging the battery whenever the system is on, and switching the RAM to battery power when the system goes off or the power fails. According to Macrotech data, it takes about 1 hour of charge to supply 150 hours of back-up power, and a full battery will keep on supplying power for at least 50 days. This means that for systems that are used daily, or even just 2 or 3 days per week, the charge-discharge cycle is totally automatic.

I've worked on this article for 3 days, keeping my file in the V-RAM Macro-Drive. Although I copied the file to a real disk each day as a precautionary measure, when I switched on the machine each following morning, there was my file still intact in the V-RAM. Even nicer was being able to leave a copy of all the files associated with the C compiler and several utilities permanently stored in the V-RAM. There they remain, always at your fingertips, and ready for quiet, lightning-fast operation. Now that the V-RAM is available, small hard disks are obsolete (well, almost obsolete; eight V-RAMS still cost a lot more than a 4-Megabyte hard disk with controller).

I have been pointing out the advantages of the battery back-up when the V-RAM is in Macro-Drive mode. However, the back-up scheme can be equally useful for those wanting to use the board primarily as system memory. With appropriate software, one could for example keep the operating system permanently loaded on the board and boot from it every time the computer is turned on.

In addition to the on-board-battery back-up scheme, the V-RAM contains circuitry to allow the back-up power (~ 2.3 volts DC) to come from an external battery through S-100 pin 21 (Note: Pin 21 is an S-100 bus uNDEFined pin. With a minor modification, the V-RAM can instead accept external back-up power from any other uNDEFined pin). An external battery could be useful to extend the duration of back-up.

The V-RAM is also capable of lowering the RDY, PWRFAIL*, or NMI* lines of the S-100 bus to signal the CPU when the board's source of power, the +8-volt line, is unreliable for proper operation. Which of these three lines is driven is jumper-selectable.

Other Features

The V-RAM consists of a 5-layer PC host board and a 4-layer adapter board. Only the host board resides on an S-100 slot, and the attached adapter board does not interfere with the use of the next slot on the bus. The boards are populated with low-power 8K×8 Static CMOS RAM chips(HM6264LP-12).

When I first examined the V-RAM, I was disappointed to see that the RAM chips had no sockets. This of course makes it very difficult for a user to change a bad chip. I called Macrotech to find out the company's philosophy on not using sockets. Well, it seems that a large percentage of the units returned for repair (of an earlier RAM board) had failed because of bad contacts between the chips and the sockets. So they decided to eliminate the problem. Macrotech points out that the V-RAM boards are burned-in for 48 hours, that the board carries a 1-year warranty, and that it is unlikely that any chip will fail after the warranty period. Although I am still not totally convinced, the argument does seem valid.

In any case, I am impressed by Macrotech's direct support of its customers. Besides offering a 1-year warranty on the V-RAM, the company is willing to answer questions that a user might have about the board or about installing it. After the warranty is over, if for any reason the board fails, it can be returned directly to Macrotech for repair. Most out-of-warranty repairs are $50 to $100, and usually include an upgrade to the latest board version.

INSTALLING AND USING V-RAM

A set of twenty-five easily-accessible box jumpers is used to select the various options, address, and I/O ports of the board (see Table 2).

Selecting the Base Address

To use the V-RAM as system memory, you must first select a base address as explained earlier. Then you must set jumpers J4 through J11 to that address. This is done as follows: Each of these jumpers represents one bit, for a total of eight bits. If a jumper is set to the left, the value of the corresponding bit is 0; if the jumper is set to the right, the value of the bit is 1. The pattern of 0's and 1's defined by the eight jumpers represents an 8-bit binary number. That binary number represents the last 64K **after** which the address starts.

Here is an example to clarify what I just said: Suppose that you already have 256K of memory and that you want the V-RAM to start immediately afterwards. Since 256K contains

4 times 64K, your V-RAM is to start immediately **after** the 4th 64K. Thus change 4 into an 8-bit binary number, arriving at

0 0 0 0 0 1 0 0.

So, the jumpers

J4-J5-J6-J7-J8-J9-J10-J11

should be set to

left-left-left-left-left-right-left-left.

In addition, the V-RAM manual contains a table showing the jumper settings for all possible base addresses.

Some operating systems automatically detect how much memory is available in the computer (provided that all boards have the addresses set right). For others, it is necessary to run a small program, or procedure, that configures a copy of the operating system for the required memory size. Consult your OS manual and installation notes to see if you need to reconfigure the system to the new memory size after you add the V-RAM.

Selecting the Port Address

To use the V-RAM as RAM-disk, the address of the first I/O port must be set correctly. This is done with jumpers J24 through J18. If the jumper is up, it corresponds to 0; if the jumper is down, it corresponds to 1. Again, the objective is to obtain the address as an 8-bit number. Note that there are only 7 jumpers; the least significant bit (the rightmost digit in a binary number) has no jumper to represent it, being always 0.

Example: For CompuPro systems, the V-RAM-disk first I/O port must be set to 0C6 (hexadecimal), which is

1 1 0 0 0 1 1 0

in binary. So the seven jumpers

J24-J23-J22-J21-J20-J19-J18

should read

1 1 0 0 0 1 1

or, in other words

down-down-up-up-up-down-down.

The address of the second I/O port is always consecutive to the first, so it needs no jumper setting.

Independently of the port address, you must tell the system how many V-RAM-disks (up to eight) are in your computer by assigning each one a number. The first board is board 0, the second board 1, etc. Then use jumpers J3-J1 to assign each board its number in binary (right=0). For example, the fourth board is board 3, or 011 in binary, so set jumpers J3-J2-J1 to right-left-left.

JUMP	FUNCTION	SETTING
J1-J3	Selects nth RAM-disk board	See text
J4-J11	Selects base address	See text
J12	Sets board as system memory	Right = Enabled
J13	Sets board as RAM-disk	Right = Enabled
J14	Pull-up resistor for PHANTOM* line	Left@ = Connected to bus
J15	PHANTOM* recognition	Right@ = Enabled
J16	Pull-up resistor for sXTRQ* line	Left@ = Connected to bus
J17	External (via bus) battery	Up@ = Disconnected from bus
J18-J24	Selects port address	See text

J25	Selects wait or interrupt signal	Open@ = not connected to bus
		Up = Drives line 12 (NMI*)
		Right = Drives line 13 (PWRFAIL*)
		Down = Drives line 72 (RDY)

Table 2. *The jumpers of the V-RAM board and their functions. Typical settings for jumpers J14-J17 and J25 are indicated by @. For setting the other jumpers see text.*

Software Drivers for RAM-Disk

If you have an operating system supplied by CompuPro (CP/M 2.2, CP/M 816, CP/M-68K, Concurrent DOS 816), you should already have the software to drive M-Drive/H. The same software will drive the V-RAM. If you have an older version of a CompuPro operating system without M-Drive software, you can usually get an upgrade (for a charge) from a CompuPro System Center. Under a CompuPro environment, the V-RAM first needs to be formatted with a program called MFORM; it then becomes the M: 'disk' drive. Drive M: is used by the user and the operating system exactly like drives A: and B:, but you will love the speed and silence.

For those with other systems, V-RAM driver software needs to be written and patched into the operating system, and a formatting program written. The article by Leonard Schwab (in this issue) demonstrates how this can be done for CP/M 2.2. The principles should hold for other operating systems. (If anyone has written or is planning to write V-RAM, or M-Drive/H, drivers for any other operating system, please contact me. S-100 Journal would be interested in publishing your results.)

WRAP-UP

Because of its dual-purpose capability and clever battery back-up scheme, the V-RAM is one of the best buys in S-100 Static RAM that I presently know. Add to that Macrotech's full 1-year warranty and the company's commendable willingness to support the user directly, and you've got a deal that you truly cannot refuse.

For more information on the V-RAM, write or send an Editorial Feature Reply Card to:

MACROTECH
9551 Irondale Avenue
Chatsworth, CA 91311
800-824-3181
818-700-1501 (Calif.)

Listing continued from page 45

```
;       SET PARAMETERS FOR EVERY RAM-DISK ACCESS OPERATION
;
        MVI C,  SECLEN                  ;BYTES TO BE MOVED = 1 SECTOR
        LHLD    DMAADR                  ;[HL] = MEMORY SOURCE OR DEST'N
        LDA     DSKCMD                  ;GET I/O OPERATION CODE
        ANI     RDCMD                   ; IF NOT READ
        JZ      PUTRAM                  ;    THEN DO WRITE OPERATION
                                        ;    ELSE FALL TO READ OP'N
;
;       MOVE [C] BYTES FROM RAM-DISK TO >>[HL]
;
GETRAM: IN      HDATA                   ;GET BYTE FROM M-DRIVE
        MOV M,  A                       ;STORE IN MEMORY
        INX H                           ;NEXT BYTE
        DCR C                           ;COUNT DOWN
        JNZ     GETRAM                  ;... TILL FINISHED
        XRA A
        RET                             ;ALWAYS RETURNS ZERO - SUCCESS
;
;       MOVE [C] BYTES FROM >>[HL] TO RAM-DISK
;
PUTRAM: MOV A,  M                       ;BYTE TO STORE
        OUT     HDATA                   ; .. TO RAM-DISK
        INX H                           ;NEXT BYTE
        DCR C                           ;COUNT DOWN
        JNZ     PUTRAM                  ;... TILL FINISHED
        XRA A
        RET                             ;ALWAYS RETURNS ZERO - SUCCESS
;
;END RAM-DISK-I/O SECTION
;------------------------------------------------------------------------
;
;START COLD-BOOT SECTION
;
;       EXTERNAL SUBROUTINES:
;               MOVCPM, lo$mem - WARM-BOOT SECTION
;
CBOOT:  lxi sp, 100h                    ;temporary stack
        call    hwinit                  ;initialize hardware as needed
;
        LDA     WRCMD                   ;WRITE TO RAM-DISK ...
        CALL    MOVCPM                  ; FROM HI MEMORY
;
        call    sgnon                   ;display sign-on message
        call    lo$mem                  ;initialize low-memory
        xra     a                       ;
        mov c,  a                       ;let CCP set default disk A:
        jmp     ccp                     ;enter ccp - AUTOLOAD active
;
hwinit:     ;put hardware initialization here
        ret
;
sgnon:      ;put routine to display sign-on message here
        ret
;
;END COLD-BOOT SECTION
;------------------------------------------------------------------------
;
```

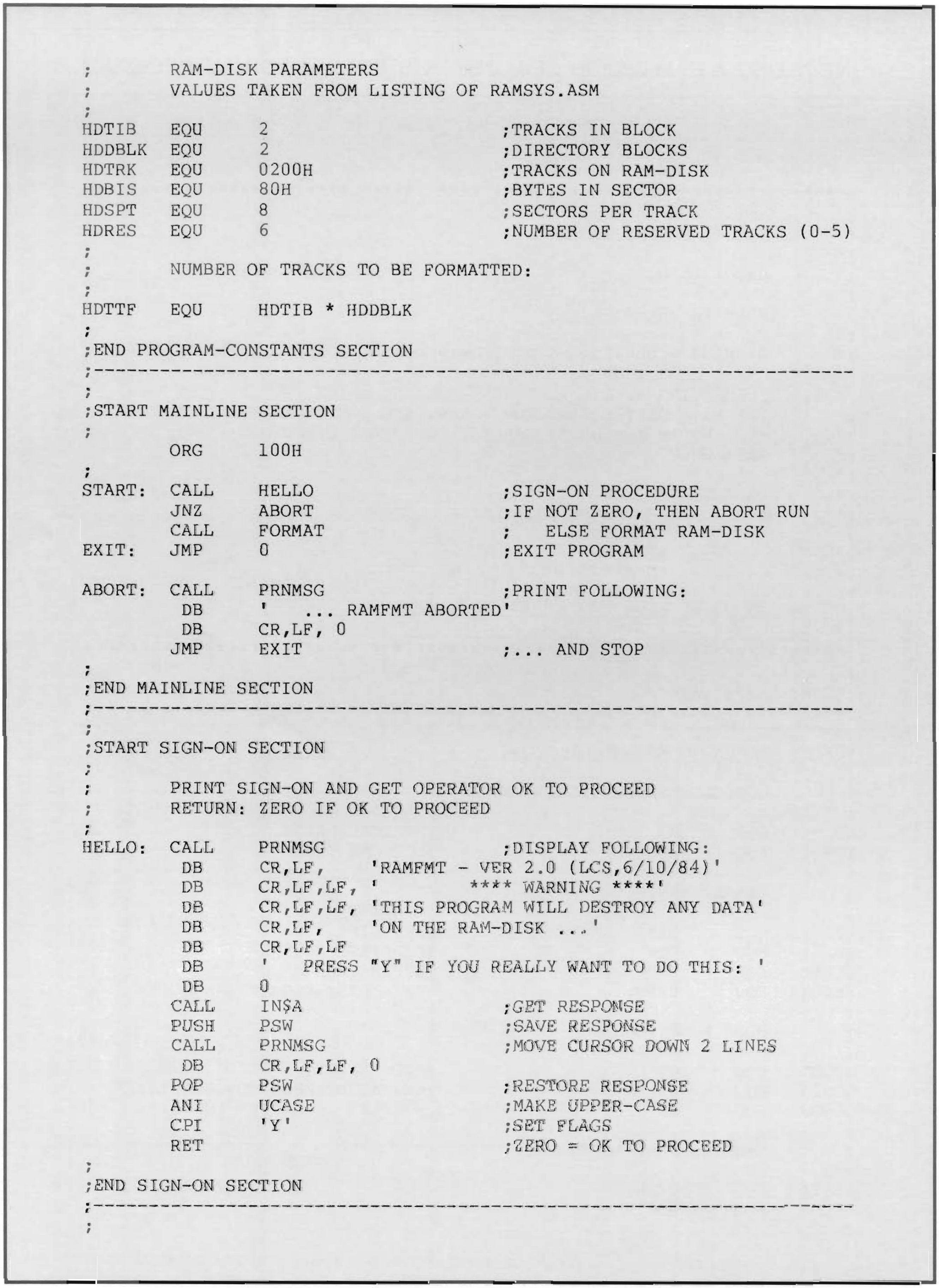
```
;START UNINITIALIZED-DATA-AREA SECTION
;
;       FOLLOWING CODE REDEFINES COLD-BOOT SECTION
;       FOR USE BY OS AFTER COMPLETION OF COLD-BOOT
;
        ORG     CBOOT
;
COUNT:  DW      0                   ;USED BY MOVCPM ROUTINE
;
;       DISK ACCESS DATA AREA
;
DISK:   DB      0                   ;DRIVE SELECTED
DSKCMD: DB      0                   ;DISK OPERATION CODE
TRACK:  DW      0                   ;TRACK-NUMBER
SECTOR: DB      0                   ;SECTOR-NUMBER
DMAADR: DW      0                   ;>> DISK I/O BUFFER
ALVRAM: DS      HDDSM+1             ;ALLOCATION-VECTOR AREA FOR RAM-DISK

DIRBUF: DS      SECLEN              ;COMMON BUFFER FOR DISK DIRECTORIES
;
ENDLOAD:
;
BIOSLEN EQU $ - BIOS                ;LENGTH OF THIS MODULE
;
;
;END UNINITIALIZED-DATA-AREA SECTION
;-----------------------------------------------------------------------
```

INSTALLING A RAM-DISK IN CP/M 2.2 — LISTING 3 — INITIALIZE RAM-DISK

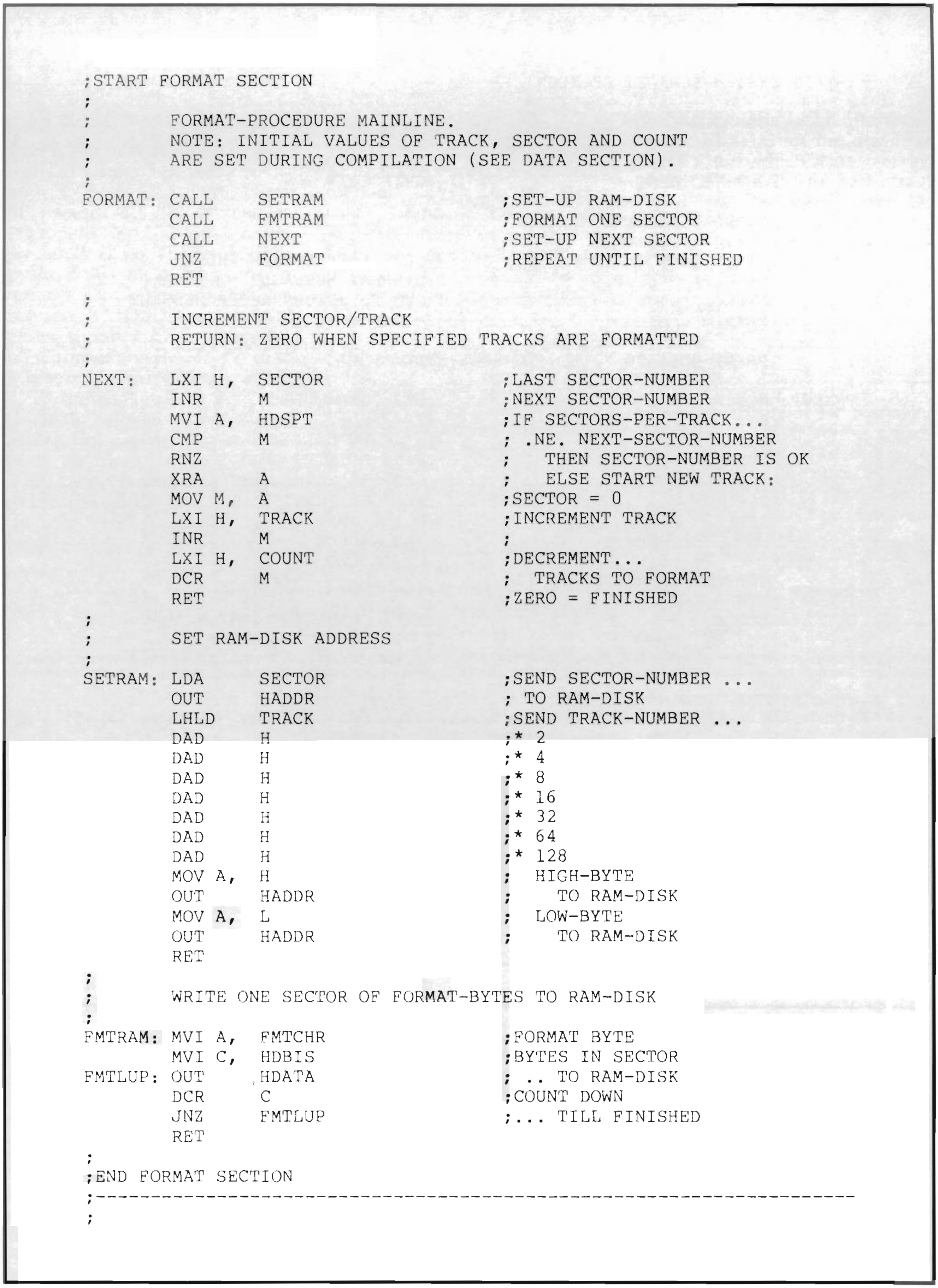
```
;RAM.SUB - SUBMIT FILE TO INITIALIZE RAM-DISK
;
;RUN RAM FORMATTING PROGRAM:
ramfmt
;
;CREATE NULL TRANSIENT PROGRAM, IN ORDER TO REPEAT PIP COMMAND
save 0 @.com
;
;TRANSFER UTILITIES TO RAM-DISK
pip c:=pip.com
@ c:=sub.com
;etc
;
;LOAD WORKING BIOS - CONTROL WILL TRANSFER TO NEW BIOS
; AND RAM:DISK WILL BE A:
ramasys
;
;END RAM.SUB
```

INSTALLING A RAM-DISK IN CP/M 2.2 — LISTING 2 — FORMAT PROGRAM

```
;*********************************************************************
;*
;*      title: RAMFMT.ASM
;*
;*      date: 15NOV85                   by Leonard Schwab
;*
;*      FORMATS RAM-DISK
;*
;*      WARNING - ERASES ALL DICTIONARY-DATA ON RAM-DISK
;*
;*      FORMATTING STARTS AT TRACK = HDRES, WHICH IS
;*      THE BASE OF THE DIRECTORY-AREA, AND CONTINUES
;*      FOR (HDTIB * HDDBLK) TRACKS, TO FORMAT DIRECTORY-
;*      AREA ONLY.
;*
;*      RAMFMT.ASM SECTIONS:
;*               PROGRAM-CONSTANTS
;*               MAINLINE
;*               SIGN-ON
;*               FORMAT
;*               SYSTEM-SUPPORT
;*               DATA
;*
;*********************************************************************
;
;START RAMFMT.ASM
;---------------------------------------------------------------------
;
;START PROGRAM-CONSTANTS SECTION
;
;       LOGICAL AND PHYSICAL CONSTANTS:
;
NO      EQU     0                     ;LOGICAL FALSE
YES     EQU     NOT NO                ;LOGICAL TRUE
;
;       CHARACTERS:
;
LF      EQU     0AH                   ;NEW LINE
CR      EQU     0DH                   ;CARRIAGE-RETURN
UCASE   EQU     5FH                   ;UPPER-CASE MASK
FMTCHR  EQU     0E5H                  ;CP/M DISK-FORMAT CHARACTER
;
;       BDOS FUNCTIONS
;
BDOS    EQU     05                    ;BDOS ENTRY
CONIN   EQU     01                    ;CHARACTER FROM CONSOLE IN [A]
CONOUT  EQU     02                    ;DISPLAY CHARACTER IN [E]
;
;       M-DRIVE PORT ADDRESSES
;
HDATA   EQU     0C6H
HADDR   EQU     HDATA + 1
;
```



```
;       RAM-DISK PARAMETERS
;       VALUES TAKEN FROM LISTING OF RAMSYS.ASM
;
HDTIB   EQU     2                     ;TRACKS IN BLOCK
HDDBLK  EQU     2                     ;DIRECTORY BLOCKS
HDTRK   EQU     0200H                 ;TRACKS ON RAM-DISK
HDBIS   EQU     80H                   ;BYTES IN SECTOR
HDSPT   EQU     8                     ;SECTORS PER TRACK
HDRES   EQU     6                     ;NUMBER OF RESERVED TRACKS (0-5)
;
;       NUMBER OF TRACKS TO BE FORMATTED:
;
HDTTF   EQU     HDTIB * HDDBLK
;
;END PROGRAM-CONSTANTS SECTION
;----------------------------------------------------------------------
;
;START MAINLINE SECTION
;
        ORG     100H
;
START:  CALL    HELLO                 ;SIGN-ON PROCEDURE
        JNZ     ABORT                 ;IF NOT ZERO, THEN ABORT RUN
        CALL    FORMAT                ;   ELSE FORMAT RAM-DISK
EXIT:   JMP     0                     ;EXIT PROGRAM

ABORT:  CALL    PRNMSG                ;PRINT FOLLOWING:
         DB     '   ... RAMFMT ABORTED'
         DB     CR,LF, 0
        JMP     EXIT                  ;... AND STOP
;
;END MAINLINE SECTION
;----------------------------------------------------------------------
;
;START SIGN-ON SECTION
;
;       PRINT SIGN-ON AND GET OPERATOR OK TO PROCEED
;       RETURN: ZERO IF OK TO PROCEED
;
HELLO:  CALL    PRNMSG                ;DISPLAY FOLLOWING:
         DB     CR,LF,    'RAMFMT - VER 2.0 (LCS,6/10/84)'
         DB     CR,LF,LF, '        **** WARNING ****'
         DB     CR,LF,LF, 'THIS PROGRAM WILL DESTROY ANY DATA'
         DB     CR,LF,    'ON THE RAM-DISK ...'
         DB     CR,LF,LF
         DB     '   PRESS "Y" IF YOU REALLY WANT TO DO THIS:  '
         DB     0
        CALL    IN$A                  ;GET RESPONSE
        PUSH    PSW                   ;SAVE RESPONSE
        CALL    PRNMSG                ;MOVE CURSOR DOWN 2 LINES
         DB     CR,LF,LF, 0
        POP     PSW                   ;RESTORE RESPONSE
        ANI     UCASE                 ;MAKE UPPER-CASE
        CPI     'Y'                   ;SET FLAGS
        RET                           ;ZERO = OK TO PROCEED
;
;END SIGN-ON SECTION
;----------------------------------------------------------------------
;
```



```
;START FORMAT SECTION
;
;       FORMAT-PROCEDURE MAINLINE.
;       NOTE: INITIAL VALUES OF TRACK, SECTOR AND COUNT
;       ARE SET DURING COMPILATION (SEE DATA SECTION).
;
FORMAT: CALL    SETRAM                ;SET-UP RAM-DISK
        CALL    FMTRAM                ;FORMAT ONE SECTOR
        CALL    NEXT                  ;SET-UP NEXT SECTOR
        JNZ     FORMAT                ;REPEAT UNTIL FINISHED
        RET
;
;       INCREMENT SECTOR/TRACK
;       RETURN: ZERO WHEN SPECIFIED TRACKS ARE FORMATTED
;
NEXT:   LXI H,  SECTOR                ;LAST SECTOR-NUMBER
        INR     M                     ;NEXT SECTOR-NUMBER
        MVI A,  HDSPT                 ;IF SECTORS-PER-TRACK...
        CMP     M                     ; .NE. NEXT-SECTOR-NUMBER
        RNZ                           ;   THEN SECTOR-NUMBER IS OK
        XRA     A                     ;   ELSE START NEW TRACK:
        MOV M,  A                     ;SECTOR = 0
        LXI H,  TRACK                 ;INCREMENT TRACK
        INR     M                     ;
        LXI H,  COUNT                 ;DECREMENT...
        DCR     M                     ;  TRACKS TO FORMAT
        RET                           ;ZERO = FINISHED
;
;       SET RAM-DISK ADDRESS
;
SETRAM: LDA     SECTOR                ;SEND SECTOR-NUMBER ...
        OUT     HADDR                 ; TO RAM-DISK
        LHLD    TRACK                 ;SEND TRACK-NUMBER ...
        DAD     H                     ;* 2
        DAD     H                     ;* 4
        DAD     H                     ;* 8
        DAD     H                     ;* 16
        DAD     H                     ;* 32
        DAD     H                     ;* 64
        DAD     H                     ;* 128
        MOV A,  H                     ;  HIGH-BYTE
        OUT     HADDR                 ;    TO RAM-DISK
        MOV A,  L                     ;  LOW-BYTE
        OUT     HADDR                 ;    TO RAM-DISK
        RET
;
;       WRITE ONE SECTOR OF FORMAT-BYTES TO RAM-DISK
;
FMTRAM: MVI A,  FMTCHR                ;FORMAT BYTE
        MVI C,  HDBIS                 ;BYTES IN SECTOR
FMTLUP: OUT     HDATA                 ; .. TO RAM-DISK
        DCR     C                     ;COUNT DOWN
        JNZ     FMTLUP                ;... TILL FINISHED
        RET
;
;END FORMAT SECTION
;----------------------------------------------------------------------
;
```

```
;START SYSTEM-SUPPORT SECTION
;
;       CONSOLE INPUT
;
IN$A:   MVI C,  CONIN                   ;READ CONSOLE ...
        JMP     BDOS                    ; VIA CP/M
;
;       DISPLAY IN-LINE MESSAGE
;
PRNMSG: XTHL                            ;GET >>MESSAGE IN [HL]
        CALL    PRNLUP                  ;DISPLAY MESSAGE
        XTHL                            ;SET TO RETURN AFTER MESSAGE
        RET
;
;       PRINT MESSAGE >>[HL] TO NULL-BYTE
;
PRNLUP  MOV A,  M                       ;GET CHARACTER
        ORA A                           ;IF ZERO ...
        RZ                              ;   THEN EXIT
        PUSH    H                       ;SAVE POINTER
        CALL    OUT$A                   ;DISPLAY CHARACTER
        POP     H                       ;RESTORE POINTER
        INX     H                       ;POINT NEXT
        JMP     PRNLUP                  ;AND REPEAT
;
;       DISPLAY CHARACTER IN [A]
;
OUT$A:  MOV E,  A                       ;PASS CHARACTER [A]
        MVI C,  CONOUT                  ;  TO CONOUT FUNCTION
        JMP     BDOS                    ;  IN CP/M
;
;END SYSTEM-SUPPORT SECTION
;---------------------------------------------------------------------
;
;START DATA SECTION
;
TRACK:  DW      HDRES
SECTOR: DB      00
COUNT:  DW      HDTTF                   ;ONLY LOW-BYTE IS USED
;
;END DATA SECTION
;---------------------------------------------------------------------
        END                             ;RAMFMT.ASM
```