



North Star Computers, Inc.
1440 Fourth Street, Berkeley, CA 94710 USA
(415) 527-6950 TWX/Telex (910) 366-7001

**SYSTEM SOFTWARE MANUAL
ADDENDUM**

July 1980
Revision 2.1

Horizon is a registered trademark of North Star Computers, Inc.

Copyright © 1979, 1980, by North Star Computers, Inc.
All Rights Reserved

Part Number 25501E

North Star Computers, Inc.
July 11, 1980

IMPORTANT: If your new Release 5.2 diskette will not boot up in your computer, read the section of this addendum on the MOVER program. You may need to use that program first.

This addendum describes the new features which have been added to North Star System Software since Release 5.0, which is described in Revision 2.1 of the SYSTEM SOFTWARE MANUAL.

SYSTEM SOFTWARE MANUAL ERRATA

Before proceeding, please make the following changes to the manual:

1. On page A-2 of the DOS section, file directory entry bytes 1C-11 are described as, "number of blocks in file", but should be described as, "number of sectors in file".
2. On page G-2 of the DOS section, one of the comments before DCOM reads, "ACC=NUMBER OF BLOCKS". It should read, "ACC=NUMBER OF SECTORS".
3. After a call to DCOM, the stack pointer will be left unchanged whether DCOM exited via the return address on the stack or via the HDERR vector. If this information might ever be useful to you, please make a note of it on page G-2 of the DOS section.
4. When performing DOS personalization, it is important to note that DOS 5.2DQ loads in two parts:

Sectors 4-8 load into 100H-AFFH (assuming normal origin)
Sectors 8-9 load into A00H-DFH

For example, if the personalization begins by typing "LF DOS 4000", then an I/O subroutine destined to be loaded at 2934H, or at 0A34H, will be found at 4834H, and a turnkey command string destined to be loaded at 2A55H, or at 0B55H, will be found at 4955H. Please make a note of this on page F-2 of the DOS section.

NEW FEATURES IN RELEASE 5.2

This section of this addendum describes the new features which were added with Release 5.2. This information is presented separately for the convenience of current users of the preceding release of the software.

As before, there are two versions of Release 5.2. One version, 5.2S, is for use with the older, single-density systems only. The other version, 5.2DQ, is for use with double-density and quad systems, only. The DOS's for these two systems are quite different because they operate different disk hardware, and some of the differences are noted below. But the remainder of the software is independent of the type of system (single-density or double-density/quad) on which it is used.

"MOVER" RELOCATION PROGRAM

The MOVER program provides a simple method for relocating DOS, BASIC, and other programs, to special starting addresses. This procedure can be easily performed directly by the dealer or end user. (Special relocated versions of DOS and BASIC will no longer be available from North Star.)

In order to provide this program relocation capability, the Release 5.2 software diskette includes the MOVER program and a series of relocation key files. For each program on the diskette, which can be relocated, there is also a corresponding relocation key file. A relocation key file has type 3, and the file name is the name of the program to be relocated preceded by a hyphen, "-".

Programs on the software diskette are relocated by LOADING and RUNNING the MOVER program under BASIC. When MOVER is run, it will make requests for relocation information on the console terminal. Then, it will sequentially attempt to relocate each program on the diskette for which a relocation key file exists.

If the relocation of a program on the diskette is not desired, the corresponding relocation key file should be deleted from the diskette before MOVER is run.

WARNING: In order to save space, both in memory, and on the disk, this MOVER program writes the newly generated, relocated program into the same files from which the relocation key information is taken. This can only be done once. Therefore, it is very important that this only be done to diskette copies made from the Factory Master diskette, and never to the original Factory Master diskette, itself.

It may be necessary to use the MOVER program before any of the other new programs can be used at all. This will be the case in a system with a nonstandard bootstrap PROM set. This will also be the case in a double-density or quad system with no memory below 2000H, because the DOS, as supplied for these systems, loads at 100H. In these cases, MOVER should be run under an earlier release of North Star BASIC.

After the relocation is completed, and before the newly generated software can be used, it will be necessary to copy the new DOS into the first file position (sector 4) of a diskette, where it can be loaded by the bootstrap PROM. It might be appropriate to copy all of the relocated software to another diskette, starting with the DOS.

When using MOVER, it is possible to supply relocation information which leads to unreasonable or meaningless program relocations. In such cases, MOVER will cause either an OUT OF BOUNDS ERROR or the generation of unusable software (such as with DOS and BASIC overlapping). When this happens, do not be alarmed. Just recopy the diskette and repeat the process until you are completely satisfied with the results. Some of the mistakes which are expected to be the most common are described below.

The utilities are always followed in RAM by large areas (5 to 5.5 K) which are used for disk buffers. Like the programs, these areas may not wrap around the end of memory. Therefore, an attempt to locate the utilities above the space occupied by the standard disk controller (E800H-EBFFH) will fail.

As mentioned above, the MOVER works by comparing the relocation key files with the corresponding standard files on the same disk. Therefore, if one of the standard files has been personalized or otherwise altered, it will not relocate properly.

When an error occurs during the relocation of one of the files, the first part of the relocation key file will have been overwritten, but its type will still be 3. This file should be recopied from the master diskette right away. If instead MOVER is run again, it will attempt to use the overwritten portion of this file as relocation key data. This could produce very obscure results.

DOS

I/O Device Status Routines

I/O device status check routines are now included in the user's I/O area to facilitate real time interactive and background operations. However, these routines are not used by any current software. Therefore, it is not necessary, at this time, for customized I/O systems to be regenerated with this feature included. The status routines are called with a device number in the accumulator and return the number of the device actually tested, if any, also in the accumulator. The input status check at 2041H or 141H, returns the Z flag true if the specified device has input data available. The output status check, at 2044H or 144H, returns the Z flag true if the specified device is ready to receive more data without delay. All other registers must be preserved by both routines. When these routines are not implemented, their positions in the jump table should contain the three byte sequence, AF 3D C9, which returns a "not ready" indication and an illegal device number (-1).

Interrupts During Disk Transfers

Interrupts are now disabled during disk transfers and conditionally reenabled when permissible. Bit 7 of the RWCHK flag byte now controls this feature, while bit 0 of the same byte continues to control the read-after-write check. Interrupts will be left enabled after disk operations if and only if bit 7 has been set to one. This does not permit interrupts to serve as effectively as the OFTEN call for such things as type ahead, software clocks, or modem service because of the length of time that they must be disabled. But for extremely rare events, such as RAM parity errors, they may be appropriate.

Parity Errors

A routine is provided, when the DOS is personalized for the Horizon, which prints a message at the console if a memory failure is ever detected. To enable this feature, the RAM boards should be set to generate vectored interrupt five and the interrupt control flag, bit 7 of RWCHK, in the working copy of the DOS should be set to one. If the message, "RAM PARITY ERROR," appears on the terminal, you may press RETURN to continue with the parity check disarmed or press any other key to reboot (after putting the appropriate diskette into drive one).

DLOOK Error Return

The library routine, DLOOK, now returns a zero in the accumulator if an error was detected in the syntax of the specified filename. Otherwise, it returns the implied drive number, as it did in earlier releases.

Disk Controller Address

There is now a byte in the jump table at 203AH or 13AH, which indicates the origin of the disk controller. This information is provided for use by any program which may need to access the disk controller directly, such as to reboot the system by branching to the address of the bootstrap PROM.

DOS FOR DOUBLE-DENSITY AND QUAD

Speed Improvement

The low level reading and writing routines in the DOS have been modified for an increase in overall throughput. Some speed improvements will be noticed during copy operations and during extensive access to data files.

CK UTILITY

The CK Utility provides an easy way for the dealer or end user to verify that the contents of a Factory Master diskette are precisely correct, and have not been corrupted or accidentally changed, after leaving the factory. Running this utility computes an overall check code for an entire diskette. The number it displays can then be compared against the correct check value for any standard diskette, as given below. If the check value does not compare correctly, then the diskette should be replaced with a correct diskette.

The correct check values for the four standard diskettes are:

<u>Name</u>	<u>Part Number</u>	<u>Check Value</u>
HORIZON-S	11002C	21516
HORIZON-DQ	11001B	58777
MDS-S	11008C	54115
MDS-DQ	11007B	38179

BASIC

Longer Maximum Line Length and Suppression of Automatic Carriage Returns

The LINE statement can now be used to set the line length for any device to as much as 165 characters. This statement will also now accept an optional, additional argument, which must be a numeric expression preceded by a comma. If this expression evaluates to zero, BASIC will stop automatically sending carriage returns and line feeds to the specified output device whenever its line length is reached. If the expression evaluates to a non-zero value, automatic generation of carriage returns and line feeds will be resumed on the specified device. This feature may be useful when sending something other than normal text to an output device.

File Size Function

A new function, FILESIZE(N), returns the size in blocks of a currently open file specified by the numeric expression, N.

File Pointer Function

Similarly, another new function, FILEPTR(N), returns the current position of the file pointer in currently open file, N. This function may be useful in the numeric expression which specifies the new position in a file for a random READ or WRITE statement.

File Pointer Positioning

A WRITE statement of the following form:

```
WRITE #N %P, NOENDMARK
```

will set the file pointer in any currently open file N, to any valid position P, for a subsequent sequential READ or WRITE statement. This statement may be used to reset a file pointer to zero without closing and reopening the file.

RAMTEST

Two copies of a new memory test program are provided, which occupy different areas of memory so that all of memory can be tested. This test is designed for use in Horizons only, requires a Z-80 processor, and does all of its I/O through the Horizon's standard serial port.

RAMTEST always tests memory in 1K regions. It maintains a 64 x 8 matrix of memory error status. Whenever this matrix is typed out, the 64 characters per line represent the 64K address space, left to right, starting at zero. The line for bit 0 is typed first, the line for bit 7 last. The error matrix is cumulative. This means that once a bit is marked bad, it will not be marked good by subsequent passes.

Phase 0 is only performed once. It makes a quick judgment of the status of the memory system by testing only the first byte of each region. It then types the error matrix, with entries meaning:

M = RAM memory
 - = no memory
 P = PROM or ROM
 * = region reserved for test program

On subsequent passes only the "M" regions are tested. Errors change the "M" to "?".

Phase 1 tests with a pattern of all zeroes.

Phase 2 tests with a pattern of all ones.

Phase 3 tests with a fixed pattern of 19 bytes repeated for the entire region. The test is done 38 times, with each byte of the pattern getting to be first, and incrementing memory both forward and backward.

Phase 4 tests with a pattern calculated to stress address buffers.

Phase 5 tests that a data pattern in each region survives stores to all other "M", "-", or "P" regions.

Phase 6 tests with an algorithm calculated to stress the ability of dynamic RAM cells to retain information in the face of noise in adjacent cells.

Phase "M" tests the ability of each region to execute programs. First a subroutine of 1023 NOP's and a RET is called several times. Next, a program calculated to create electrical noise is called, which moves itself about while performing calculations. If the correct answer is returned, an "M" is typed. If the wrong answer is returned, a "?" is typed. If the subroutine fails to return, most likely nothing is typed, leaving the terminal's cursor indicating the offending region.

Phase "-" tests a memory board's ability to fetch an opcode, operand sequence from different memory chips, testing that the first chip deselected properly. This phase is performed only on even numbered passes to ensure that the computer first executes all other numbered phases.

Each full pass begins by typing out a pass counter which runs from "AA" to "ZZ" and then repeats. The following character is an exclamation point if any errors have occurred since the test began, or a space otherwise.

If a control C from the keyboard is detected (through the Horizon's standard serial port), the program jumps to location E800h to reboot. If any other character is detected, the error matrix is typed. Thus evidence of a single error is not lost, even if it has scrolled off of a CRT terminal.

The program assumes the presence of a North Star dual-density disk controller, which will generate wait states whenever location E918H is fetched. Obviously, the program will still run without wait states. A fetch of location E918H has no effect on a single-density controller. Wait states are used in phases 6 and "M".

For North Star RAMS, parity control is output on port COH. Phase 0 attempts to light the error LED if any byte had bad parity on entering the program. After phase 0, the LED should light only on the occurrence of an error. A scan for correct parity is also done at the end of each phase. A chip location diagram for North Star RAM boards is included at the end of this addendum.

Z80 Horizon I/O Routines

The I/O routines for Horizon computers supplied with Release 5.2 are written in Z80 code. This was done to make room for the parity message feature and to make it usable in systems which do not have RAM at zero. However, except for the RAMTEST, the remainder of the software on a Release 5.2 diskette is still entirely 8080 compatible.

3

```

0000      ;*
0000      ;*I/O ROUTINES FOR STANDARD HORIZON COMPUTER
0000      ;*
0000      ;*
0000      ;*
0000      ;*
0000      ;
0100      DOS      EQU      0100H ; BASE ADDRESS OF DOS
0000      ;
0000      ;
0000      PADDR EQU      0 ; BASE ADDRESS OF MOTHERBOARD I/O PORTS
E800      BADDR EQU      0E800H ; BASE ADDRESS OF DISK CONTROLLER
0000      ;
0000      ;
0000      P0       EQU      PADDR+0 ;ADDRESSES OF MOTHERBOARD I/O PORTS
0001      P1       EQU      PADDR+1
0002      P2       EQU      PADDR+2
0003      P3       EQU      PADDR+3
0004      P4       EQU      PADDR+4
0005      P5       EQU      PADDR+5
0006      P6       EQU      PADDR+6
0007      P7       EQU      PADDR+7
0000      ;
0000      ;
0A00      IOBLK  EQU      DOS+900H
0107      OFTEN  EQU      DOS+7
012B      RWCHK  EQU      DOS+2BH
0000      ;
0000      ;
0000      ORG      DOS+0DH
010D      C3DA0A  CHO    JMP      COUT ; CHARACTER OUTPUT
0110      C3500A  CHI    JMP      CIN ; CHARACTER INPUT
0113      C3880A  INIT   JMP      TINIT ; INITIALIZATION
0116      C3F10A  CON    JMP      CONTC ; CONTROL C CHECK
0119      ;
0119      ;
0119      ;
0119      ORG      DOS+41H
0141      ;
0141      ; THIS IS THE INPUT STATUS ROUTINE
0141      ; IT IS CALLED WITH THE DEVICE # IN A
0141      ; RETURNS NUMBER OF DEVICE TESTED IN A
0141      ; RETURNS Z FLAG TRUE IF INPUT DATA AVAILABLE
0141      ; NO OTHER REGISTERS MAY BE MODIFIED
0141      C33A0A  ISTAT  JMP      IST
0144      ;
0144      ; THIS IS THE OUTPUT STATUS ROUTINE
0144      ; IT IS CALLED WITH THE DEVICE # IN A
0144      ; RETURNS NUMBER OF DEVICE TESTED IN A
0144      ; RETURNS Z FLAG TRUE IF OUTPUT DEVICE READY
0144      ; NO OTHER REGISTERS MAY BE MODIFIED
0144      ; ISTAT AND OSTAT MAY BE USED BY SOFTWARE TO
0144      ; DETERMINE WHICH DEVICE NUMBERS ARE IMPLEMENTED

```

```

0144 C3680A   OSTAT JMP   OST
0147         ;
0147         ;
0147         ;
0147         ;           ORG   IOBLK
0A00         ;
0A00         ; THE FOLLOWING ROUTINE IS CALLED IF A PARITY
0A00         ; ERROR EVER OCCURS AND GENERATES AN INTERUPT.
0A00         ; (PE JUMPERED TO VI5 ON THE RAM BOARD)
0A00         ; IT GIVES THE USER AN OPPORTUNITY TO ATTEMPT TO
0A00         ; CONTINUE PROCESSING BY PRESSING RETURN OR TO
0A00         ; REBOOT BY STRIKING ANY OTHER KEY.
0A00 PERR    EQU    $
0A00 F5      PUSH   PSW
0A01 C5      PUSH   B
0A02 E5      PUSH   H
0A03 21200A  LXI    H,ERTXT      ;POINT TO ERROR MESSAGE
0A06         PERR1  EQU    $
0A06 46      MOV    B,M
0A07 23      INX    H
0A08 AF      XRA    A           ;SPECIFY OUTPUT DEVICE ZERO
0A09 CD0D01  CALL   CHO           ;SEND CHARACTER
0A0C B7      ORA    A           ;TEST FOR TERMINATING NULL
0A0D 20F7    JRNZ   PERR1
0A0F CD1001  CALL   CHI           ;GET RESPONSE FROM DEVICE ZERO
0A12 FE0D    CPI    13        ;TEST FOR CARRIAGE RETURN
0A14 3E40    MVI    A,40H
0A16 D3C0    OUT    0C0H       ;DISARM PARITY LOGIC
0A18 C200E8  JNZ    BADDR       ;REBOOT UNLESS CHAR WAS CARRIAGE RETURN
0A1B E1      POP    H
0A1C C1      POP    B
0A1D F1      POP    PSW
0A1E FB      EI
0A1F C9      RET
0A20         ;
0A20 0D0A5241 ERTXT  DB    13,10,'RAM PARITY ERROR ',7,0
         4D205041
         52495459
         20455252
         4F522007
         00
0A35         ;
0A35         ;
0A35         ;
0A35 OFIST  EQU    $           ;ALTERNATIVE ENTRY TO STATUS ROUTINE
0A35 F5      PUSH   PSW       ;SAVE DEVICE NUMBER
0A36 CD0701  CALL   OFTEN      ;TAKE CARE OF BUSINESS
0A39 F1      POP    PSW       ;RESTORE DEVICE NUMBER
0A3A         IST    EQU    $           ;INPUT STATUS ROUTINE
0A3A FE01    CPI    1         ;TEST FOR DEVICE 1 POSSIBILITY
0A3C 2808    JRZ    IST1       ;JUMP TO SECOND SERIAL PORT STATUS TEST
0A3E         ;* ASSUME DEVICE 0 WAS INTENDED
0A3E         IST0  EQU    $
0A3E DB03    IN     P3         ;FIRST SERIAL STATUS PORT
0A40 2F      CMA           ;INVERT STATUS FOR PROPER RESULT
0A41 E602    ANI    2         ;TEST RECEIVER DATA AVAILABLE BIT

```

```

0A43 3E00          MVI    A,0      ;SHOW WHICH DEVICE WAS TESTED
0A45 C9           RET     ;RETURN WITH INPUT STATUS IN Z FLAG
0A46             ;*
0A46             IST1  EQU    $
0A46 DB05         IN     P5
0A48 2F          CMA
0A49 E602        ANI    2
0A4B 3E01        MVI    A,1
0A4D C9          RET
0A4E             ;*
0A4E             ZCIN  EQU    $      ;ALTERNATIVE ENTRY TO CIN
0A4E 3E00        MVI    A,0      ;SUBSTITUTE FIXED DEVICE NUMBER
0A50             CIN   EQU    $      ;CHARACTER INPUT ROUTINE
0A50 CD3A0A      CALL   IST      ;CHECK STATUS OF SPECIFIED DEVICE
0A53 20FB        JRNZ  CIN      ;LOOP UNTIL DATA AVAILABLE
0A55 FE01        CPI    1        ;CHECK FOR DEVICE 1 POSSIBILITY
0A57 2805        JRZ   CIN1     ;JUMP IF SECOND SERIAL PORT SPECIFIED
0A59             ;*ASSUME PORT 0 (STANDARD SERIAL PORT) DESIRED
0A59             CINO  EQU    $
0A59 DB02        IN     P2        ;INPUT THE CHARACTER
0A5B E67F        ANI    7FH      ;MASK OFF PARITY BIT
0A5D C9          RET     ;RETURN WITH CHARACTR IN A
0A5E             ;
0A5E             CIN1  EQU    $
0A5E DB04        IN     P4
0A60 E67F        ANI    7FH
0A62 C9          RET
0A63             ;
0A63             ;
0A63             OFOST EQU    $      ;ALTERNATIVE ENTRY TO STATUS ROUTINE
0A63 F5          PUSH  PSW      ;SAVE DEVICE NUMBER
0A64 CD0701      CALL  OFTEN   ;TAKE CARE OF BUSINESS
0A67 F1          POP   PSW      ;RESTORE DEVICE NUMBER
0A68             OST   EQU    $      ;OUTPUT STATUS ROUTINE
0A68 FE02        CPI    2        ;TEST FOR DEVICE 2 POSSIBILITY
0A6A 2814        JRZ   OST2     ;JUMP TO PARALLEL PORT STATUS TEST
0A6C FE01        CPI    1        ;TEST FOR DEVICE 1 POSSIBILITY
0A6E 2808        JRZ   OST1     ;JUMP TO SECOND SERIAL PORT STATUS TEST
0A70             ;* ASSUME DEVICE 0 WAS INTENDED
0A70             OST0  EQU    $
0A70 DB03        IN     P3        ;FIRST SERIAL STATUS PORT
0A72 2F          CMA      ;INVERT STATUS FOR PROPER RESULT
0A73 E601        ANI    1        ;TEST TRANSMITTER BUFFER EMPTY BIT
0A75 3E00        MVI    A,0      ;SHOW WHICH DEVICE WAS TESTED
0A77 C9          RET     ;RETURN WITH INPUT STATUS IN Z FLAG
0A78             ;*
0A78             OST1  EQU    $
0A78 DB05        IN     P5
0A7A 2F          CMA
0A7B E601        ANI    1
0A7D 3E01        MVI    A,1
0A7F C9          RET
0A80             ;*
0A80             OST2  EQU    $
0A80 DB06        IN     P6
0A82 2F          CMA

```

```

0A83 E601      ANI    1
0A85 3E02      MVI    A,2
0A87 C9        RET
0A88          ;
0A88          ;
0A88          TINIT EQU    $
0A88          ;* FIRST INITIALIZE MOTHERBOARD AND SET UP BOTH SERIAL PORTS
0A88 AF        XRA    A          ;ZERO ACC
0A89 D306      OUT    P6          ;INITIALIZE MOTHERBOARD
0A8B D306      OUT    P6          ;EXTRA
0A8D D306      OUT    P6          ;EXTRA
0A8F D306      OUT    P6          ;EXTRA
0A91 3ECE      MVI    A,0CEH     ;2 STOPS, 16xCLOCK, 8 BITS, NO PARITY
0A93 D303      OUT    P3          ;SEND TO FIRST SERIAL PORT
0A95 3ECE      MVI    A,0CEH     ;SAME CODE AS FIRST PORT
0A97 D305      OUT    P5          ;SECOND PORT
0A99 3E37      MVI    A,37H     ;CMD: RTS, ER, RXF, DTR, TXEN
0A9B D303      OUT    P3          ;FIRST PORT
0A9D 3E37      MVI    A,37H     ;SAME CODE AS FIRST PORT
0A9F D305      OUT    P5          ;SECOND PORT
0AA1          ;*
0AA1          ;*
0AA1          ;*TINIT NEXT REWRITES ALL RAM TO SET PARITY CORRECT
0AA1          ;*
0AA1 3E40      MVI    A,40H     ;DISABLE PARITY LOGIC
0AA3 D3C0      OUT    0C0H      ;BEFORE READING UNWRITTEN RAM
0AA5 2100E8    LXI    H,BADDR     ;FIRST BYTE TO CLEAR
0AA8 54        MOV    D,H
0AA9 5D        MOV    E,L
0AAA 0101FC    LXI    B,-1023        ;NUMBER OF BYTES TO CLEAR
0AAD EDB8      LDDR     ;SET PARITY ON ALL RAM
0AAF 3C        INR    A          ;TO 41H, PARITY ENABLE CODE
0AB0 D3C0      OUT    0C0H      ;REARM PARITY LOGIC
0AB2          ;
0AB2 DB02      IN     P2          ;CLEAR STANDARD SERIAL PORT INPUT BUFFER
0AB4 DB04      IN     P4          ;CLEAR SECOND SERIAL PORT INPUT BUFFER
0AB6          ;*
0AB6          ;*
0AB6 060D      MVI    B,13          ;CARRIAGE RETURN TO INIT PRINTER
0AB8 3A2B01    LDA    RWCHK
0ABB 07        RLC          ;TEST INTERUPT ENABLE FLAG
0ABC 3007      JRNC   COUT2
0ABE 3E0A      MVI    A,IOBLK/256    ;PAGE ADDRESS OF I/O BLOCK
0AC0 ED47      MOV    IV,A
0AC2 ED5E      IM     2          ;SET INTERUPT MODE TWO
0AC4 FB        EI
0AC5          ;
0AC5          ;*PRINTER PARALLEL OUTPUT ROUTINE
0AC5          COUT2 EQU    $
0AC5 3E20      MVI    A,20H
0AC7 D306      OUT    P6          ;RESET PO FLAG
0AC9 78        MOV    A,B          ;CHARACTER TO SEND
0ACA F680      ORI    80H         ;SET STROBE FALSE
0ACC D300      OUT    P0          ;SEND CHARACTER
0ACE EE80      XRI    80H         ;TOGGLE STROBE
0AD0 D300      OUT    P0

```

```

0AD2 EE80      XRI    80H      ;TOGGLE STROBE
0AD4 D300      OUT     P0
0AD6 78       MOV     A,B     ;GET CHARACTER FOR RETURN
0AD7 C9       RET
0AD8          ;
0AD8          ;
0AD8          ZCOUT EQU    $      ;ALTERNATIVE ENTRY TO COUT
0AD8 3E00      MVI     A,0     ;SUBSTITUTE FIXED DEVICE NUMBER
0ADA          COUT EQU    $      ;CHARACTER OUTPUT ROUTINE
0ADA CD680A    CALL    OST     ;CHECK STATUS OF SPECIFIED DEVICE
0ADD 20FB      JRNZ   COUT    ;LOOP UNTIL READY FOR DATA
0ADF FE01      CPI     1
0AE1 2808      JRZ    COUT1   ;SECOND SERIAL PORT OUTPUT
0AE3 FE02      CPI     2
0AE5 28DE      JRZ    COUT2   ;PARALLEL OPORT OUTPUT
0AE7          ;*ASSUME STANDARD SERIAL PORT OUTPUT
0AE7          COU0 EQU    $
0AE7 78       MOV     A,B     ;MOVE CHARACTER TO A
0AE8 D302      OUT     P2     ;OUTPUT THE CHARACTER
0AEA C9       RET
0AEB          ;
0AEB          COUT1 EQU   $
0AEB 78       MOV     A,B
0AEC D304      OUT     P4
0AEE C9       RET
0AEF          ;
0AEF          ;
0AEF          ;
0AEF          DS     IOBLK+0EFH-$
0AEF          ; MODE TWO INTERRUPT VECTOR FOR RESTART FIVE
0AEF 000A      DW     PERR
0AF1          ;
0AF1          CONTC EQU   $
0AF1 3E00      MVI     A,0     ;MAIN CONSOLE DEVICE NUMBER
0AF3 CD3A0A    CALL    IST     ;TEST STATUS OF CONSOLE
0AF6 37       STC
0AF7 3F       CMC     ;ENSURE CARRY FALSE
0AF8 C0       RNZ   ;RETURN IF NO CHARACTER TYPED
0AF9 CD500A    CALL    CIN     ;INPUT THE CHARACTER THAT WAS FOUND AVAILABLE
0AFC FE03      CPI     3     ;SEE IF CHARACTER IS CONTROL-C
0AFE 37       STC     ;TELL SOFTWARE A CHAR WAS TYPED
0AFF C9       RET     ;RETURN WITH Z-FLAG PROPERLY SET
0B00          ;

```

SYMBOL TABLE

BADDR	E800 00	CHI	0110 01	CHO	010D 01	CIN	0A50 01	CIN0	0A59 01	CIN1	0A5E 01
CON	0116 01	CONTC	0AF1 01	COU0	0AE7 01	COUT	0ADA 01	COUT1	0AEB 01	COUT2	0AC5 01
DOS	0100 00	ERTXT	0A20 01	INIT	0113 01	IOBLK	0A00 00	IST	0A3A 01	IST0	0A3E 01
IST1	0A46 01	ISTAT	0141 01	OFIST	0A35 01	OFOST	0A63 01	OFTEN	0107 00	OST	0A68 01
OST0	0A70 01	OST1	0A78 01	OST2	0A80 01	OSTAT	0144 01	P0	0000 00	P1	0001 00
P2	0002 00	P3	0003 00	P4	0004 00	P5	0005 00	P6	0006 00	P7	0007 00
PADDR	0000 00	PERR	0A00 01	PERR1	0A06 01	RWCHK	012B 00	TINIT	0A88 01	ZCIN	0A4E 01
ZCOUT	0AD8 01										

+

BUGS FIXED IN RELEASE 5.2

DOS

The CR command now fails properly when given an illegal filename.

The read-after-write check option now works with the IN command.

Hard disk error messages are now routed to device 0.

DOS for Single-Density

After "PRESS RETURN TO CONTINUE" appears in a directory list, device 0 is now specified in the call to CIN.

After RETURN is pressed to continue, a carriage return and a line feed are now echoed to the console.

The auto start option can now be set up even without using a fresh copy of the DOS.

When the auto start feature is used, the DOS herald is not displayed.

DOS for Double-Density and Quad

The DOS will now boot up properly regardless of the previous contents of memory at its load address.

Directory listings will no longer cause the system to hang up for an undue length of time in the case of uninitialized or poorly mounted diskettes.

Directory listings also will not damage any software which may have overlaid the latter part of the DOS, starting at 2A00H.

OFTEN is now called during seeks on drives with fast track-to-track stepping.

CO Utility

The proper way to compact overlapping files has never been agreed upon. Therefore, CO will no longer offer to proceed in the presence of conflicts.

CO will no longer offer to convert a purely double density diskette to double-density.

Density mismatch errors will now be handled properly in all cases.

CF Utility

Small files can now be copied into very large files of 512 blocks or more.

Like the CR command, CF will now fail to create a new file given an illegal filename.

Like the CO utility, CF will now handle density mismatches properly in all cases.

BASIC

The obscure bug involving the CREATE statement and its affect on several other, seemingly unrelated statements has now been fixed.

Numeric overflows will now be detected properly when dividing very large numbers by very small numbers, even without a Hardware Floating Point Board.

Use of user defined functions in the THEN clause of an IF statement will no longer prohibit the use of an ELSE clause in the same statement.

The DEF statement can now be followed by additional statements on the same line, specifically, the problem of following a DEF statement with one containing an equal sign, "=", has now been solved.

NEW RELEASE 5.1 FEATURES

This section describes the new features incorporated with Release 5.1 system software.

The format of a double sided diskette is as follows: Side A is the same as a single sided diskette and can be used in single sided disk drives. Side B holds tracks 35 through 69, with track 35 at the inside (opposite track 34), and with track 69 at the outside (opposite track 0). The directory is still on side A, but entries in it may refer to disk addresses up through 699. On a double sided, double-density diskette, a file which does not overlap the directory area may now be as large as 1392 blocks (384K bytes, or 696 sectors). Double sided diskettes may be used in single sided drives, but only the data on side A may be accessed. If a file "wraps around" to side B, only the part of it on side A can be accessed in a single sided drive. Any attempt to access the remainder of such a file will cause an error.

Single density DOS 5.1S now supports the CD, CF, CO, and DT utility programs, and has all the features of the double-density DOS except the OFTEN call and, of course, double-density or double sided operation. In the single-density DOS, note that the JMP instruction at 2007H or 0107H has nothing to do with OFTEN and must not be altered.

The CD, CF, CO, and DT utility programs included with Release 5.1 system software are not the same as those that were supplied with Release 5.0 and will not work properly with that or any earlier DOS. These utilities will now support double sided operations when used with the DOS 5.1DQ.

There is one new personalization byte, CONFIG, at 2034H or 0134H in standard versions of Release 5.1 DOS. It tells the DOS and other system software two things about each of the four possible disk drives in the system:

1. The upper four bits tell which drives are double sided. Bits 7 down through 4 correspond to drives 1 through 4, respectively. Each bit must be 0 if the corresponding drive is single sided, or 1 to allow double sided operation of that drive.
2. North Star's new, double sided drives are capable of much faster stepping between tracks. The lower four bits of CONFIG tell the DOS which drives have this feature. Bits 0 through 3 correspond to drives 1 through 4, respectively. Each bit must be 0 if the corresponding drive can only step at normal speed, or 1 to take advantage of fast-stepping.

For example, a CONFIG value of 5AH (01011010) indicates that only drives 2 and 4 are double sided, fast stepping drives. The proper CONFIG value for four quad capacity drives is OFFH. DOS on factory master diskettes is supplied with zero in the CONFIG byte, because any drive can be operated at normal speed, using side A only.

Note that the CONFIG byte exists in both single density DOS 5.1S and dual density DOS 5.1DQ. The CONFIG byte in the single density 5.1S must never be changed from its original zero value, because the single density system supports neither double sided diskette access nor fast-stepping.

The personalization process for DOS Release 5.1S is the same as that for Release 4 DOS, except that the jump table, after and including address 200DH, corresponds to the description of the Release 5.0 jump table in the SYSTEM SOFTWARE MANUAL. In particular, such features as the PAGES byte, the AUTOSTART flag, and a pointer to the command buffer, which have been in the double-density DOS jump table since release 5.0, are now contained within the jump table for the single-density DOS, 5.1S, as well.

The procedure for personalizing the dual density DOS remains the same as that given in the SYSTEM SOFTWARE MANUAL, except that the proper value for CONFIG should be set in the new copy of DOS (in the workspace RAM) before it is saved on the personalized diskette.

Beginning with Release 5.1, DOS on factory master diskettes will have the read-after-write option enabled.

Two diskettes are provided with all standard HORIZON computer systems and MDS Micro Disk Systems: a FACTORY MASTER DISKETTE containing system software, and a blank to be used in preparing the WORKING DISKETTE as described in the SYSTEM SOFTWARE MANUAL. The blank included with quad-capacity systems is certified for full double sided operation, but the Factory Master system software diskette for ALL systems will remain single sided only, because recording upon the factory master, or removing its write protect tab, should never be done.

Standard Release 5.1 BASIC is now configured to use 24K of RAM, starting at address 2000H or 0100H. (This now includes over 8K for a user program and data.) Previous releases were configured for 16K of RAM.

Users of FPBASIC in single-density systems should note that the standard address of the floating point board was changed from DFF0H to EFFF0H as of system software Release 5.0. At this time, a new personalization byte, FPBADDR (ORG + 21H), was added to FPBASIC (not regular BASIC). Changing FPBADDR enables FPBASIC to make use of a floating point board with address other than EFFF0H. See DISCUSSION: PERSONALIZING BASIC in the SYSTEM SOFTWARE MANUAL for further details.

CHIP LOCATOR FOR NORTH STAR RAM 16:

0000	2000	4000	6000	8000	A000	C000	E000
14D-14C	14B-14A	14D-14C	14B-14A	14D-14C	14B-14A	14D-14C	14B-14A
13D-13C	13B-13A	13D-13C	13B-13A	13D-13C	13B-13A	13D-13C	13B-13A
15D-15C	15B-15A	15D-15C	15B-15A	15D-15C	15B-15A	15D-15C	15B-15A
16D-16C	16B-16A	16D-16C	16B-16A	16D-16C	16B-16A	16D-16C	16B-16A
11D-11C	11B-11A	11D-11C	11B-11A	11D-11C	11B-11A	11D-11C	11B-11A
10D-10C	10B-10A	10D-10C	10B-10A	10D-10C	10B-10A	10D-10C	10B-10A
12D-12C	12B-12A	12D-12C	12B-12A	12D-12C	12B-12A	12D-12C	12B-12A
17D-17C	17B-17A	17D-17C	17B-17A	17D-17C	17B-17A	17D-17C	17B-17A
-----	MMMM**MM	-----	14B-14A	14D-14C	-----	-----	--P-----
-----	MMMM**MM	-----	13B-13A	13D-13C	-----	-----	--P-----
-----	MMMM**MM	-----	15B-15A	15D-15C	-----	-----	--P-----
-----	MMMM**MM	-----	16B-16A	16D-16C	-----	-----	--P-----
-----	MMMM**MM	-----	11B-11A	11D-11C	-----	-----	--P-----
-----	MMMM**MM	-----	10B-10A	10D-10C	-----	-----	--P-----
-----	MMMM**MM	-----	12B-12A	12D-12C	-----	-----	--P-----
-----	MMMM**MM	-----	17B-17A	17D-17C	-----	-----	--P-----
	^ good board		^ board under test				

CHIP LOCATOR FOR NORTH STAR RAM 32:

0000	2000	4000	6000	8000	A000	C000	E000
--14D---	--14C---	--14B---	--14A---	--14D---	--14C---	--14B---	--14A---
--13D---	--13C---	--13B---	--13A---	--13D---	--13C---	--13B---	--13A---
--15D---	--15C---	--15B---	--15A---	--15D---	--15C---	--15B---	--15A---
--16D---	--16C---	--16B---	--16A---	--16D---	--16C---	--16B---	--16A---
--11D---	--11C---	--11B---	--11A---	--11D---	--11C---	--11B---	--11A---
--10D---	--10C---	--10B---	--10A---	--10D---	--10C---	--10B---	--10A---
--12D---	--12C---	--12B---	--12A---	--12D---	--12C---	--12B---	--12A---
--17D---	--17C---	--17B---	--17A---	--17D---	--17C---	--17B---	--17A---
-----	MMMM**MM	-----	--14A---	--14D---	--14C---	--14B---	--P-----
-----	MMMM**MM	-----	--13A---	--13D---	--13C---	--13B---	--P-----
-----	MMMM**MM	-----	--15A---	--15D---	--15C---	--15B---	--P-----
-----	MMMM**MM	-----	--16A---	--16D---	--16C---	--16B---	--P-----
-----	MMMM**MM	-----	--11A---	--11D---	--11C---	--11B---	--P-----
-----	MMMM**MM	-----	--10A---	--10D---	--10C---	--10B---	--P-----
-----	MMMM**MM	-----	--12A---	--12D---	--12C---	--12B---	--P-----
-----	MMMM**MM	-----	--17A---	--17D---	--17C---	--17B---	--P-----
	^ good board		^ board under test				

NOTES:

1. Each column of the RAMTEST table represents 1K of address space.
2. The columns are grouped by eights; each group corresponds to one of the selection switches on a North Star RAM board.
3. The first row printed is for bit 0, the last is for bit 7.
4. Meaning of designators:
 - M = good memory
 - ? = questionable memory
 - P = PROM/ROM
 - = no memory