NorthStar™

System Software Manual

25013B
Revision 2.1

North Star Computers, Inc.
14440 Catalina St., San Leandro, CA 94577 USA
(415) 357-8500 TWX/Telex (910) 366-7001

SYSTEM SOFTWARE MANUAL

REVISION 2.1

Part Number 25013B

PREFACE

This manual describes all the system software that is included with a North Star HORIZON® computer or Micro Disk System. Use of the North Star Disk Operating System (DOS), Monitor, and BASIC are described in three of the major sections of this manual. The first major section, GETTING STARTED, describes the initial procedure required to begin using the North Star software.

The table of contents for all the major sections of this manual follows this preface. Two indexes for the BASIC section appear at the very end of the manual. If you receive errata sheets for this manual, be sure to incorportate all the corrections into the manual, or attach the errata sheets to the manual.

This manual applies to North Star system software diskettes stamped "RELEASE 5" or "RELEASE 5.X", where X is a digit indicating the update number. If you are working with earlier releases of North Star software, you should order a copy of the most recent release to take full advantage of all the features described in this manual. This manual covers both single-density and double-density versions of the North Star software. Differences between single- and double-density versions are noted in the text.

Other software available for your North Star system is not described here. For example, North Star Pascal and the North Star Software Exchange diskettes are not described here. Consult a North Star Catalog, Newsletter, or your local computer dealer for up-to-date descriptions of available North Star software.

Every effort has been made to ensure the accuracy of the material presented here. Nevertheless, experience shows that some textual errors always go undetected. If you find any errors, or have some suggestions on how to improve this manual, please contact North Star at the following address:

NORTH STAR COMPUTERS, INC.
ATTN: SOFTWARE DOCUMENTATION
14440 CATALINA STREET
SAN LEANDRO, CA 94577 USA

IV. THE NORTH STAR BASIC SYSTEM (Continued)

IV. THE NORTH STAR BASIC SYSTEM (Continued)

GETTING STARTED

INTRODUCTION

This part of the manual provides the information and procedures required to make initial use of the North Star system software. This material should be referenced at any of the following times:

A. You are about to use an assembled HORIZON computer or MICRO DISK System for the first time.

B. You have just finished assembling and checking a HORIZON or MICRO DISK System from kit.

C. You are about to use a new release system software diskette for the first time.

The sections that follow provide:

A. Information on the disk drives and how to use them.

B. Itemization of the system software provided with North Star disk systems.

C. Procedures for personalizing the DOS software to make possible input/output communication with your computer's console terminal.

D. Procedures for testing your computer's RAM memory and disk system for correct and reliable operation.

These sections should be read carefully and the specified procedures should be followed in the order given.

DISK DRIVES AND DISKETTES

Your North Star HORIZON or MICRO DISK System equipped computer includes capability for storing large amounts of data and program information on "floppy" diskettes. There may be up to four floppy disk drives connected to your computer through one disk controller board (only three drives for single density controllers). Looking at the front of a disk drive, you will see a small red LED indicator lamp, a slot running through the center of the face, and hinged door at the center, perpendicular to the slot. When the door is closed, no diskette may be inserted into the disk drive. Opening the door permits the withdrawal of the diskette and insertion, through the slot, of another diskette. When the LED light is on, it indicates that that drive is active. The disk system incorporates an automatic shut off feature which will turn off the drive motor(s) when not in use to save wear on both diskettes and disk drives.

DESCRIPTION OF DISKETTES

A diskette is a magnetically coated, thin plastic disk which is permanently sealed within a square protective jacket. The label on the jacket should be in the upper left corner as you are looking at the diskette. There are three holes in the front face of the jacket. The large hole in the middle allows the disk drive spindle to clamp directly onto the diskette in order to spin it around. Data is stored onto and retrieved from diskette much as with a phonograph record, except that, for a diskette, the needle is a magnetic record/playback head. The small, round hole to the lower right of the diskette is called the sector detect hole and is of no importance to this discussion. The large oblong hole at the bottom and the corresponding hole on the flip side expose the diskette's magnetic surface for the record/playback heads. The little square notch in the upper right corner of the diskette is a write protect notch. If you cover this notch with an adhesive tab, then the disk drive will be inhibited from writing over the information stored on the diskette. It will be read-only until the write protect tab is removed at which time both reading and writing of the diskette will be possible.

INSERTION OF DISKETTES

When you insert the diskette into the disk drive, be sure that you are holding the label edge of the diskette, and that it slides all the way in, oblong hole first, with the label facing away from the drive's LED indicator. In a HORIZON, the write protect notch should be at the top. In a horizontally oriented disk drive the notch should be at the left. After the diskette is inserted, make sure the door on the drive is locked into the closed position before you attempt to use the diskette.

CARE OF DISKETTES

Diskettes are delicate and should be handled with great care. Always observe the following rules in the handling and storing of diskettes.

1. Never directly touch the magnetic surfaces of a diskette.
2. Never bend or fold a diskette.
3. Keep a diskette in its protective envelope when not in use.
4. Never expose a diskette to heat, X-ray or other radiation, magnetic fields, moisture, or dust.

LIST OF SYSTEM SOFTWARE PROGRAMS

The following programs are included on a North Star system software diskette:

DOS	The Disk Operating System program.
CO	Utility program for compacting a diskette and optionally converting a diskette to double density.
CD	Copy diskette utility program.
CF	Copy file utility program.
DT	Disk test utility program.
BASIC	The BASIC language system program with software arithmetic.
FPBASIC	The BASIC language system program set up for use with the hardware floating point board.
M2D00	The Monitor program with origin 2D00 (hex). (M2A00 if single density diskette)
M5700	The Monitor program with origin 5700 (hex) and built-in HORIZON input/output routines.
M6700	The Monitor program with origin 6700(hex).
M0000	The Monitor program with origin 0.
MF400	The Monitor program with origin F400 (hex).

RAM ALLOCATION

The following table shows how the 64K byte RAM address space is allocated for the standard version system software and hardware. All addresses are given in hexadecimal notation. The minimum memory configuration requires 16K of RAM in the address range 2000-5FFF(hex).

SINGLE DENSITY	DOUBLE DENSITY	PROGRAMS
2000-29FF	2000-2CFF	DOS
2A00-5FFF *	2D00-5FFF *	BASIC, FPBASIC
2A00-31FF	2D00-34FF	Monitor M2A00, M2D00
2A00-3AFF	2D00-47FF	Utilities CO, CD, CF, DT
5700-5FFF	5700-5FFF	Monitor M5700
6700-6EFF	6700-6EFF	Monitor M6700
E800-EBFF	E800-EBFF	Disk Controller
EFF0-EFFF **	EFF0-EFFF **	Floating Point Board
F400-FBFF	F400-FBFF	Monitor MF400

* The upper limit of BASIC can be set by the user with the MEMSET command. It is initially set to 5FFF.

** Some floating point boards are configured to use DFF0-DFFF.

PERSONALIZING THE DOS FOR INPUT/OUTPUT

Before the North Star system software can be used, input/output routines may have to be installed in the DOS program to allow communication of the DOS with the console terminal of your computer system. This is called "personalizing" the input/output routines of the DOS. Just exactly what steps need to be taken depends on the combination of software and hardware to be used in your system.

A. You have a HORIZON computer and the console terminal is connected to the standard serial interface. In this case, the DOS on the system software diskette supplied with the HORIZON is already personalized and ready to use. Skip to the SYSTEM START-UP section. After the system is successfully started, proceed directly to the CREATING THE WORKING DISKETTE section.

B. You have a HORIZON or other computer and you have a system software diskette which has specific input/output routines installed that match the input/output configuration of your hardware. You are ready to proceed without the need for any additional personalizing of the diskette. Skip to the SYSTEM START-UP section. After the system is successfully started, proceed directly to the CREATING THE WORKING DISKETTE section. Personalized system software diskettes for the more common input/output configuations are available. Consult the North Star Product Catalog and your dealer.

C. You have an unpersonalized system software diskette but also have a different system software diskette which is already personalized for your system. This situation might occur if you have just received a new release of the system software (unpersonalized) and wish to start using it on your already running North Star system. Proceed directly to the PERSONALIZING A NEW DISKETTE FROM AN OLD DISKETTE section.

D. You have an unpersonalized system software diskette and will install the input/output routines yourself. The MICRO DISK System is supplied with such an unpersonalized diskette. This personalization procedure is not possible unless your computer system includes some capability, such as a front panel or ROM monitor, for loading the input/ouput routines into RAM memory. Furthermore, this procedure is not simple. It requires an understanding of the computer's input/output interfaces, hexadecimal numbers, and machine language programming. If all these requirements are met, then proceed directly to the INSTALLING INPUT/OUTPUT ROUTINES section.

E. You have a HORIZON computer but will not use the standard serial interface for connecting your console terminal. In this case follow the procedure described in step D.

Also, if at any time you wish to add input/output devices to the system or modify the existing routines, you must follow the procedure described in step D.

SYSTEM START-UP

Start-up of a HORIZON computer is very simple. First, load a system software diskette into drive #1. Then, turn on the computer power. The HORIZON will automatically start the disk bootstrap program which will turn on the disk drive and load the DOS into RAM from the disk. If the computer hardware is properly configured, then the system should display a DOS command prompt (* or +) and the system will be ready to use. To do a system start-up when the power is already on, depress and release the reset switch.

In a computer system other than a HORIZON which has a North Star MICRO DISK System installed in it, start-up the system as follows. First, with no diskette loaded into any disk drives, turn on the computer and disk drive power. In some computer systems, turning the power on or off while a diskette is loaded into a drive may damage the information stored on the diskette. With the power on, load the system software diskette (already personalized) in drive #1, and then cause the computer to start executing at address E800(hex). A front panel, ROM monitor, or auto-jump feature can be used to start the computer at this address. At this point the DOS software should load as described above.

If after performing the system start-up sequence, you don't get any output on you terminal, it may be because the baud rate setting of the terminal does not match the baud rate setting of the serial interface, or it may be because of some other fault in the hardware configuration (such as improperly addressed RAM boards), or it may be some problem with the input/output personalization routines. All these possibilities should be carefully examined. If typing a key causes that character to be displayed twice, it is probably because the terminal is in half duplex mode rather than full duplex mode. If some computer operations, such as the DOS list command (LI), terminate prematurely, this may be a result of an incorrectly written control-C input/output routine. Other problems may be a result of typing lower case characters for commands instead of upper case.

PERSONALIZING A NEW DISKETTE FROM AN OLD DISKETTE

If it is desired to personalize a new system software diskette using the same personalized input/ouput routines that already exist on an old diskette, then the following procedure can be used to incorporate the old routines into the new software. The new diskette may be unpersonalized or it may have input/output routines that you wish to replace. The following listing gives the DOS and Monitor commands which should be exactly followed to copy successfully the input/output routines. The DOS command prompt will be * instead of + if a single density DOS is used. The computer must include 16K of RAM starting at 2000(hex). It is assumed that the system software has standard origin at 2000(hex).

Using an old diskette, do a system start-up sequence, then:

```
+LF DOS 4000              Load copy of old DOS
```

Next, remove the old diskette and insert a new diskette in drive #1.

```
+LF DOS 5000              Load copy of new DOS
+GO M2D00                 GO M2A00 if new diskette is single density
>MM 4900,100 5800         Move I/O routines from old to new DOS
                           (See below.)
>MM 400D,C 500D           Move jumps from old to new DOS
```

The third command in the above sequence will vary, depending upon the nature of the source and destination diskettes. To transfer old I/O personalization from Release 1, 2, 3, or 4 to Release 5 dual-density DOS, use:

```
>MM 4900,100 5800
```

as above. To transfer I/O personalization between copies of the Release 5 DOS, use:

```
>MM 4800,100 5800
```

To transfer I/O personalization between copies of single-density DOS, Release 4 or earlier, use:

```
>MM 4900,100 5900
```

If the old DOS has additional personalization, copy it now.

A copy of the new DOS with the old input/ouput routines installed now resides at address 5000(hex) in RAM. Proceed to the CREATING THE WORKING DISKETTE section for directions on how to make a diskette which includes this DOS.

INSTALLING THE INPUT/OUTPUT ROUTINES

The DOS is designed to be able to interface to any conceivable terminal input/output configuration. There are four routines required by the DOS: character input (CIN), character output (COUT), control-C detect (CONTC), and terminal initialization (TINIT). In the standard version of the DOS, the input/output routines are located in the 256 byte region from 2900 to 29FF(hex).

CIN

The purpose of CIN is to obtain a single character of input from an input device and to return the value of that character in the accumulator. When CIN is called, the accumulator will contain a device number. This value, in the range 0 to 7, specifies from which of eight possible input devices the single character of input is to be obtained. Device 0 is always assumed to be the console terminal. Devices 1 to 7 may be assigned to any other input devices in the system. CIN may be written so that it ignores the device number in the accumulator if there is only one input device in the system. CIN must do a RET to the calling routine when the input character is ready in the accumulator. The accumulator is the only register which may be modified by the CIN routine. If the input routine is complex enough to require the use of other registers, their values when CIN is called must be saved, and then restored before CIN returns.

COUT

This routine sends a single character of output information to an output device. The character to be output is provided to COUT in the B-register, and the output device number is provided in the accumulator. When COUT has finished sending the output character to the appropriate device, the character itself must be in the accumulator as well as the B-register, and the routine must do a RET back to the calling routine. No registers, other than the accumulator, may be modified by the action of the COUT routine.

CONTC

This routine detects if a control-C has been typed on the console terminal. No information is passed to the routine in any of the registers, and no registers need be saved or restored by CONTC. They are all available for unrestricted use by the routine. If a control-C has been typed, the routine should set the Zero flag. If no character has been typed or if the character typed was not a control-C, then the Zero flag should instead be cleared. As soon as the Zero flag is given its proper value, CONTC must do a RET. CONTC should not wait for a character to be typed. If no character has been typed, it should do a RET immediately after clearing the Zero flag.

TINIT

Many terminals require a special initialization procedure to be followed immediately after they are turned on for use. For example, a video display controller may require that the screen be cleared before the screen is used for the first time after power on. Also, the interface electronics (such as the HORIZON standard serial interface) may require initialization after power-on or reset. The TINIT routine is called once by the DOS right after the bootstrap load and should contain any instructions which implement this one time initialization for all input/output devices used in your system. Since many terminals do not need to be initialized, you may not need to use TINIT. TINIT may freely use all registers, without having to save or restore any. The TINIT routine should do a RET when finished.

STEP BY STEP PROCEDURE

In order to personalize the DOS with input/ouput routines for your hardware configuration, perform the following steps:

1. Write your input/ouput routines carefully following all the rules specified in the above input/output routine descriptions and the DOS ENTRY POINTS AND FLAGS section of the DOS part of this manual. As examples of correct input/output routines, the following section shows the input/ouput routines for the HORIZON.

2. Perform a system start-up sequence using the unpersonalized system software diskette. In an unpersonalized diskette, each of the input/ouput routines is set up to merely do a jump to self instruction. Thus, when you first perform a system start-up sequence, the DOS will end up in a jump to self loop in TINIT, and the unpersonalized DOS will now be loaded into RAM starting at address 2000(hex).

3. Using the computer front panel or ROM monitor, stop the computer and load your input/ouput routines into RAM in the region from 2900 to 29FF(hex).

4. Once the input/output routines have been put into computer memory, you must modify the DOS jump table so that it contains the starting addresses of each of the routines. This jump table occurs from address 200D to 2018(hex). This region is 12 bytes long. Each successive 3 byte section within it consists of an 8080/Z80 JMP instruction (C3 hex) followed by the two byte starting address (low order byte first) of one of the four routines. The following table shows how the region from 200D to 2018(hex) would be modified to recognize CIN, COUT, CONTC, and TINIT if the starting addresses for these routines were 2900, 2920, 2940, and 2960(hex), respectively.

BEFORE		AFTER		
Address	Contents	Address	Contents	
200D	C3 0D 20	200D	C3 20 29	(for COUT)
2010	C3 10 20	2010	C3 00 29	(for CIN)
2013	C3 13 20	2013	C3 60 29	(for TINIT)
2016	C3 16 20	2016	C3 40 29	(for CONTC)

Note that if TINIT is not required, the byte at 2013(hex) should be changed to a RET instruction (C9 hex).

5. If you used a front panel to modify the DOS, then the stack pointer has not been changed. So continue with execution of the new TINIT routine by causing the computer to begin execution at address 2013(hex). If you used a ROM monitor to modify the DOS, then the stack pointer may have been changed but the console terminal has been initialized by the monitor. So continue by causing the computer to begin execution at address 2028(hex), the DOS continue entry point. You should see a DOS command prompt (* or +) on your terminal. If you don't, this means that the input/output routines are faulty or a mistake was made in following the above personalization steps.

6. Copy the personalized DOS at 2000 to 5000(hex) by typing the following commands:

```
+LF DOS 5000
+GO M2D00                    GO 2A00 if single density DOS
>MM 200D,C 500D
>MM 2900,100,5800
```

(If the DOS at 5000H is not Release 5 dual-density, use:

```
>MM 2900,100 5900
```

as the last command in the above sequence, replacing the one listed.)

7. Proceed directly to the CREATING THE WORKING DISKETTE section.

```
2900            *
2900            *I/O ROUTINES FOR STANDARD HORIZON COMPUTER
2900            *     IN RELEASE 4 DOS
2900            *
2900 FE02       CIN CPI 2               CHECK FOR DEVICE 2 POSSIBILITY
2902 CA2229        JZ CIN2              JUMP IF PARALLEL PORT SPECIFIED
2905 FE01          CPI 1               CHECK FOR DEVICE 1 POSSIBILITY
2907 CA1629        JZ CIN1              JUMP IF SECOND SERIAL PORT SPECIF
290A            *ASSUME PORT 0 (STANDARD SERIAL PORT) DESIRED
290A DB03       CIN0 IN 3               INPUT FIRST SERIAL PORT STATUS
290C E602          ANI 2                MASK INPUT STATUS BIT
290E CA0A29        JZ CIN0              LOOP IF NO CHARACTER
2911 DB02          IN 2                 INPUT THE CHARACTER
2913 E67F          ANI 7FH              MASK OFF PARITY BIT
2915 C9            RET                  RETURN WITH CHARACTR IN A
2916
2916 DB05       CIN1 IN 5
2918 E602          ANI 2
291A CA1629        JZ CIN1
291D DB04          IN 4
291F E67F          ANI 7FH
2921 C9            RET
2922
2922            *SAMPLE PARALLEL INPUT CODE
2922 DB06       CIN2 IN 6            READ MOTHERBOARD STATUS
2924 E602          ANI 2             MASK TO GET THE PI FLAG
2926 CA2229        JZ CIN2           NO INPUT TYPED YET
2929 DB00          IN 0              READ DATA FROM KEYBOARD
292B F5            PUSH PSW          SAVE THE CHARACTER
292C 3E30          MVI A,30H
292E D306          OUT 6             RESET PI FLAG
2930 F1            POP PSW
2931 E67F          ANI 7FH
2933 C9            RET
2934
2934
2934 FE01       COUT CPI 1
2936 CA4929        JZ COUT1             SECOND SERIAL PORT OUTPUT
2939 FE02          CPI 2
293B CA5429        JZ COUT2             PARALLEL OPORT OUTPUT
293E            *ASSUME STANDARD SERIAL PORT OUTPUT
293E DB03       COU0 IN 3               INPUT FIRST SERIAL PORT STATUS
2940 E601          ANI 1                MASK OUTPUT STATUS BIT
2942 CA3E29        JZ COU0              LOOP IF NOT READY TO OUTPUT
2945 78            MOV A,B              MOVE CHARACTER TO A
2946 D302          OUT 2                OUTPUT THE CHARACTER
2948 C9            RET
2949 DB05       COUT1 IN 5
294B E601          ANI 1
294D CA4929        JZ COUT1
2950 78            MOV A,B
2951 D304          OUT 4
2953 C9            RET
```

```
2954                *SAMPLE PARALLEL OUTPUT ROUTINE
2954 DB06           COUT2 IN 6              READ MOTHERBOARD STATUS
2956 E601              ANI 1                MASK TO GET THE PO FLAG
2958 CA5429            JZ COUT2              PRINTER NOT YET READY
295B 78                MOV A,B              GET CHARACTER TO ACC
295C D300              OUT 0                OUTPUT TO PRINTER
295E 3E20              MVI A,20H
2960 D306              OUT 6                RESET PO FLAG
2962 78                MOV A,B           CHARACTER EXPECTED IN ACC ON RETURN
2963 C9                RET
2964
2964 DB03           CONTC IN 3              INPUT SERIAL PORT STATUS
2966 E602              ANI 2                MASK INPUT STATUS BIT
2968 EE02              XRI 2                SET Z-FLAG ONLY IF CHARACTER
296A C0                RNZ                  RETURN IF NO CHARACTER TYPED
296B DB02              IN 2                 INPUT THE CHARACTER
296D E67F              ANI 7FH              MASK OFF PARITY BIT
296F FE03              CPI 3                SEE IF CHARACTER IS CONTROL-C
2971 37                STC                  TELL SOFTWARE A CHAR WAS TYPED (OPTIONAL)
2972 C9                RET                  RETURN WITH Z-FLAG PROPERLY SET
2973
2973                *
2973                *TINIT FIRST REWRITES ALL RAM TO SET PARITY CORRECT
2973
2973 210000         TINIT LXI H,0           PREPARE TO CYCLE THROUGH RAM
2976 16E4              MVI D,BADDR/256         SET UP TO SKIP DISK REGION
2978 7C             TINKL MOV A,H           MOVE CURRENT BLOCK NUMBER TO A
2979 BA                CMP D                CHECK IF DISK BLOCK
297A C28329            JNZ TINCP            CONTINUE IF NOT DISK BLOCK
297D C604              ADI 4                ADD 1K TO RAM ADDRESS
297F 67                MOV H,A              PUT UPDATED ADDRESS BACK TO HL
2980 CA9629            JZ TINU              MAKE SURE NOT DONE IF NON-STANDARD
2983 7E             TINCP MOV A,M           READ BYTE FROM RAM
2984 77                MOV M,A              RESTORE IT WITH CORRECT PARITY
2985 2C                INR L                INCREMENT LOW ORDER ADDRESS BYTE
2986 C28329            JNZ TINCP            LOOP IF NOT AT END OF 256 BLOCK
2989 24                INR H                INCREMENT BLOCK NUMBER
298A CA9629            JZ TINU              DONE IF WE ARE BACK TO ZERO
298D 7C                MOV A,H              BLOCK NUMBER TO A
298E E603              ANI 3                MASK LOW ORDER 2 BITS
2990 C28329            JNZ TINCP            CONTINUE IF NOT AT END OF 1K BLOCK
2993 C37829            JMP TINKL            BRANCH TO MAIN LOOP
2996                *
2996                *NOW THAT ALL BYTES HAVE CORRECT PARITY, ENABLE PARITY LOGIC
2996                * (IF YOU DON'T HAVE RAM-16-A WITH PARITY, THIS IS A NOP)
2996                *
2996 3E41           TINU MVI A,41H          ENABLE PARITY CODE
2998 D3C0              OUT 0C0H             MEMORY BOARD OUTPUT PORT
299A
299A                *  NOW INITIALIZE MOTHERBOARD AND SET UP BOTH SERIAL PORTS
299A AF                XRA A                ZERO ACC
299B D306              OUT 6                INITIALIZE MOTHERBOARD
299D D306              OUT 6                EXTRA
```

```
299F D306          OUT 6           EXTRA
29A1 D306          OUT 6           EXTRA
29A3 3ECE          MVI A,0CEH      2 STOPS, 16xCLOCK, 8 BITS, NO PARIT
29A5 D303          OUT 3           SEND TO FIRST SERIAL PORT
29A7 3ECE          MVI A,0CEH      SAME CODE AS FIRST PORT
29A9 D305          OUT 5           SECOND PORT
29AB 3E37          MVI A,37H       CMD: RTS, ER, RXF, DTR, TXEN
29AD D303          OUT 3           FIRST PORT
29AF 3E37          MVI A,37H       SAME CODE AS FIRST PORT
29B1 D305          OUT 5           SECOND PORT
29B3             *
29B3             *
29B3 DB02          IN 2            CLEAR STANDARD SERIAL PORT INPUT BU
29B5 DB04          IN 4            CLEAR SECOND SERIAL PORT INPUT BUFF
29B7 3E30          MVI A,30H
29B9 D306          OUT 6           RESET PI FLAG (FOR PARALLEL PORT)
29BB C9            RET
29BC             *
```

CREATING THE WORKING DISKETTE

Before using the system software it should be copied to a diskette other than the factory supplied system software diskette. This diskette, called the WORKING DISKETTE, will be the one used on a daily basis. After this procedure is finished, the factory diskette should be retired to a safe place for storage with the write protect tab installed. If the working diskette should ever be accidentally destroyed, the factory diskette can then be used to create a new working diskette.

There are two different procedures for creating the working diskette depending on whether your computer has one or two disk drives. The procedure with two disk drives is much simpler and should be used if at all possible.

TWO DISK DRIVE PROCEDURE

Load the factory diskette with the write protect tab installed in drive #1 and a blank diskette (to become the working diskette) with no write protect tab in drive #2. Then perform the following DOS command:

```
+GO CD 1 2
```

This will copy the complete contents of the factory diskette onto the working diskette. If the factory diskette was already personalized for your system, then you are done. However, if you personalized the DOS input/output routines before coming to this section, then the personalized DOS is at 5000(hex) in RAM and should be copied to the working diskette with the following command:

```
+SF DOS.2 5000
```

Now you are done and the working diskette is ready to use.

SINGLE DISK DRIVE PROCEDURE

Load the factory diskette with write protect tab installed in drive #1 and perform the following DOS commands:

```
+RD 0 4000 8          Read file directory into RAM
+LI                   List file directory on factory diskette
```

Now remove the factory diskette and load the diskette to become the working diskette with the write protect tab removed in drive #1 and perform the following commands:

```
+IN                   Initialize working diskette
+WR 0 4000 8          Write file directory onto diskette
+LI                   List file directory
```

The listed file directory on the working diskette should be

identical to the listed file directory on the factory diskette. If you personalized the DOS input/output routines before coming to this section, then the personalized DOS is at 5000(hex) in RAM and should be copied to the working diskette with the following command:

```
+SF DOS 5000
```

Now, for each file on the factory diskette, perform a sequence like the following which copies the DOS file:

```
Load factory diskette in drive #1
+LF DOS 2D00
Load working diskette in drive #1
+SF DOS 2D00
```

The DOS file should not be copied if a personalized DOS was already copied to the diskette from RAM. After repeating the above sequence once for each file, the factory diskette will be completely copied to the working diskette. You are done and the working diskette is ready to use.

REGULAR BACKUP PROCEDURES

It is an inescapable fact that any user of a computer will make frequent mistakes in the instructions given to the computer. Most of these mistakes will be easily corrected. However, a few will cause major loss of information stored on diskettes. For example, to cite an extreme but plausible case, suppose you have spent an entire month typing a data base into your computer and it is stored on a single diskette. You now wish to initialize a new diskette and type an IN command to the DOS. It is not until the command is completed that you realize that you forgot to load the new diskette in the disk drive and that you have just initialized the diskette which held the results of one month's work. This kind of disaster can be avoided by faithfully following these two rules:

1. Always keep a write protect tab on a diskette unless you are about to write on the diskette.

2. Always make a backup copy of any file you have just changed in any significant way.

The copy disk and copy file utility programs make the backup procedure easy.

Important files or diskettes should be stored in a more permanent way. For example, a copy of the personalized working diskette should be retired to safe storage and be recovered only if the normal working diskette is destroyed.

HARDWARE TESTING

It is extremely important that you test the hardware of your computer system thoroughly using the following procedure, before using the computer for any serious work. These procedures should identify any faults or intermittent failures in the computer's RAM and disk system. These procedures should be repeated regularly in order to maintain system integrity and reliability.

RAM TEST PROCEDURE

A failure of the RAM may be the cause of almost any type of problem you may encounter while using your computer. Therefore, frequent testing of the RAM is very important. The RAM is tested with the TM command of the Monitor program. The test repeatedly writes a pattern of data into the region of RAM being tested and then reads the pattern to check that the correct pattern is indeed in the RAM. Since the test modifies the region of RAM it is testing, it is not possible to test the area where the test program itself resides. Therefore, the test procedure must be done in two steps, the first testing the last part of RAM with a Monitor program that resides in the first part of RAM and the second testing the first part of RAM with a Monitor program that resides in the last part of RAM. Start by performing a system start-up sequence and type the following command to start the Monitor:

```
+GO M2D00                    M2A00 if a single density diskette
```

With standard memory addressing, a computer with 16K of RAM will have memory in the range 2000-5FFF(hex), 32K in the range 2000-9FFF(hex), and 48K in the range 2000-DFFF(hex). Test the last part of this region with a command like the following which will test the last 8K of a 16K memory.

```
>TM 4000-5FFF 1
```

The test may run for several minutes with no apparent signs of life on the terminal. You can determine whether or not the test is still running by typing a control-C to stop the test. If the test was still running, the monitor will prompt you for another command with another (>). If nothing happens when you type the control-C, then something is wrong. If the test is allowed to run to completion, it will print the message PASS COMPLETED on the terminal and then start another pass. The program should be allowed to run for several hours to perform a thorough test.

If in the course of its operation the test detects an error in the memory it is testing, it will display on the terminal an error message of the form:

xxxx yy READ AS zz

The numbers xxxx, yy, and zz are hexadecimal. They represent the address of, the expected contents of, and the actual contents of the byte in RAM where the error was detected. If zz is always FF, then there may not be any RAM board addressed to the area being tested. Another possible cause of errors is an address conflict, for example an attempt to share the same area of memory between a RAM board and a memory mapped device, such as the disk controller or a floating point board, or another RAM board. After the test has run successfully for several hours, perform the following commands to do a similar test on the first part of RAM:

```
>OS                          Return to DOS from the Monitor
+GO M5700                    Load Monitor into last part of RAM
>TM 2000-3FFF 0
```

The M5700 Monitor has its own set of input/output routines since the RAM test will overwrite the input/output routines in the DOS at 2000(hex). Initially, the M5700 routines are personalized for use with a HORIZON. If your machine has some other hardware configuration, then the M5700 input/output routines must be changed to match your DOS routines. See the Monitor section of this manual for details. After the M5700 test has run successfully for several hours, type a control-C to stop the test and type the following command to return to the DOS.

```
>IL
```

DISK TEST PROCEDURE

In order to check for proper operation of the disk controller and disk drive(s), a DOS disk test program (DT) has been provided. This utility will repeatedly write a changing pattern to a specified drive and then attempt to read it back. Refer to the UTILITIES section of the DOS part of this manual for details on operation of the DT utility.

If each drive in your system will pass a disk test for 15 minutes, then your disk subsystem is in good operational order. If an error occurs, this may mean one of several things:

1. The diskette is improperly mounted, has a write protect tab, or has a "bad spot" which will not properly record data. If other diskettes pass the disk test, then the problem is with the diskette.

2. The disk drives are improperly connected to the system. For example, the cable connection has not been made correctly, power is not properly applied to the drives, or the drive configuration has not been done properly.

3. There is a hardware problem in the controller or drive. If your computer memory is operational, and your copy of DOS and DT have not been improperly modified, and the problem is not 1 or 2 above, then there may be a hardware problem in your disk controller or disk drive. In a multiple drive system, you can attempt to isolate the problem by testing both drives to determine if the problem is with an individual drive or not.

North Star

DISK OPERATING SYSTEM

Version 2

INTRODUCTION

The North Star DOS (Disk Operating System) was designed and implemented by staff members of North Star Computers, Inc. for use in conjunction with the North Star MICRO DISK SYSTEM, and HORIZON computer system. The DOS permits a user to issue various "commands" from a terminal for maintaining and using files on diskette. The DOS also provides "library routines" which may be called from user software. These library routines will primarily be of interest to users who will be developing their own system software, as opposed to those users who will primarily use application systems such as BASIC.

Versions of the North Star DOS are available for both single-density and double-density North Star disk systems. The DOS for single-density systems is different from the DOS for double-density systems. When reading this manual, if you have a single-density system, then ingore all references to double-density capabilities.

The DOS occupies 3.25K (D00 hex) bytes of RAM in double-density systems and 2.5K (A00 hex) bytes of RAM in single-density systems, including 256 bytes of RAM for input/output routines. No buffer area outside the DOS is required for any of the DOS commands. The origin of the DOS is 2000(hex) in both standard versions.

The North Star DOS is intended for use only with the North Star MICRO DISK SYSTEM and HORIZON computer, and no license is granted for any other use. Improved copies of the DOS, as they become available, may be obtained for a nominal charge.

Before the DOS can be used with a specific computer configuration, the instructions in the GETTING STARTED section of this manual must be followed.

ABOUT FILES

DISK ADDRESSES

Each diskette consists of 35 concentric TRACKS, and each track is subdivided into 10 SECTORS. A disk sector can hold either 512 bytes of double-density information or 256 bytes of single-density information. For purposes of discussion, a FILE BLOCK is defined to be a unit of information equal to 256 bytes. A sector can therefore contain two file blocks in double-density, or one file block in single-density. Every sector on the disk is identified by a unique DISK ADDRESS - an integer from 0 through 349. For example, sector 3 of track 27 has disk address of 273. Track 0 is the "outermost" track, and track 34 is the "innermost" track.

FILES

The primary DOS function is to permit the creation, deletion and use of files on diskettes. A file is an integral number of file blocks of data and occupies sequential disk sectors. For example, a particular file might occupy disk addresses 17 through 95 on a diskette loaded in drive #2. Note that files must always begin on sector boundaries, and that double-density files must always contain an even number of file blocks.

The first four sectors on each diskette contain a FILE DIRECTORY which specifies a symbolic name, base address, length, type, and data-density information for each file on that diskette. The symbolic name may be up to 8 characters long, and may include any characters except blank and comma. The length of a single-density file may be up to 346 blocks, and a double-density file may extend to 692 blocks. A directory may contain as many as 64 entries in single-density and 128 entries in double-density. No two files in a directory may have the same name, but it is possible for files of the same name to be in directories of diskettes loaded simultaneously on separate drives in a muliple disk drive system.

FILE TYPES

One byte in the file directory entry for each file specifies the "type" of the file. Depending on the specific type, additional bytes in the entry may have special meaning. Only four of the 127 possible file types have been assigned to date:

type 0 - Default type. All new files are assigned type 0 until explicitly changed.

type 1 - Machine language program. This file type identifies a machine language program (object code) that may be executed directly from the DOS with the GO command.

type 2 - BASIC program. This type of file is used to identify a BASIC program that can be LOADed or SAVEd from BASIC.

type 3 - BASIC data file. This type of file is the standard type for data files read and written by BASIC programs.

FILE DIRECTORY STRUCTURE

The file directory occupies disk addresses (sectors) 0 though 3. Each block in the directory holds thirty-two (sixteen in single-density systems) 16-byte entries. The symbolic name of the entry uses the first 8 bytes of an entry. An empty entry is an entry with 8 blanks (20 hex). Following the symbolic name in an entry, the disk address (2 bytes), the file size (two bytes) and the type (1 byte) follow. The last three bytes of an entry are type dependent. In particular, for a type 1 file (GO file), the two bytes following the type byte contain the go-address, and for a type 2 file (BASIC program) the byte following the type byte specifies how many file blocks of the file actually contain valid data.

File directory entry:

bytes 0-7	symbolic name of entry
bytes 8-9	disk address
bytes 10-11	number of blocks in file
byte 12	file type (high bit is 1 if double-density)
bytes 13-15	type-dependent information

COMMANDS

Instructions are issued to the DOS from the terminal by typing COMMANDS. The command format is a 2-letter mnemonic followed by any required arguments. Arguments are separated from the command mnemonic and from each other by a single blank. A command must be terminated by a carriage return before the DOS takes any action. If a typing error occurs during typing of a command, an at-sign(@) or control-N may be typed to permit re-typing of the command. Also, an underline, left-arrow, control-Q, or control-H may be typed to erase the previously typed character.

When a file name is required as a command argument, the disk drive number (in a multiple drive system) may be specified by immediately following the file name with ",1", ",2", ",3", or ",4". Drive #4 four may be specified only in double-density systems. Otherwise, drive #1 is assumed. Some sample file names are:

```
ABC    TEST1234,3    BASIC,1
```

Commands may be typed whenever the prompt character (* for single-density DOS and + for double-density DOS) appears at the left margin of the terminal.

LI <optional device specification> <optional drive number>

This command will list the entire contents of the directory on the diskette loaded in the specified drive. If no drive is specified, then drive #1 is assumed. For each file, its symbolic name, starting disk address, length, data density (single or double), and type will be printed. For type 1 files, the go-address will also be printed. To prematurely terminate a listing, a control-C may be typed. If output to a device other than the console terminal is desired, then the desired output device number may be specified by typing a # character followed by the device number. The device number must correspond to a device that has been interfaced to the system in both hardware and by adding the appropriate personalized input/output routine.

CR <file name> <length> <optional start address> <optional density>

This command will create a new file on the drive indicated by the file name. The length argument specifies the number of 256-byte blocks. If no starting address is given, then the file will start after the "last" (innermost) file currently allocated on the diskette. Otherwise, the supplied starting address will be used. The optional density specification is a single letter, "S" or "D", signifying that the file should be created in single or double density, respectively. If no density choice is specified, double-density is assumed. No density specification may be made with the single-density version of the DOS. The CR command will only create a file

directory entry - no accessing of the file itself will be done.

DE <file name>

This command will delete an existing file directory entry on the indicated drive. No actual accessing of the file blocks will be done. The DE command, in conjunction with the CR command, may be used to change the length of a file on the disk. If this is done, note that the type and type-dependent information will have to be re-entered.

TY <file name> <file type> <optional go-address>

This command is used to change the type of the specified file on the indicated drive. If type 1 is specified, then the third argument must be supplied to specify the "go-address".

GO <file name>

This command is used to load the specified file into RAM from the indicated drive and begin execution. The GO command may be used only with type 1 files. The GO command will read the entire file into RAM beginning at the go-address, and then jump to the go-address. Therefore, the first byte of the file must be the entry point of the program. The GO command sets the HL register pair to a value which points to the remainder of the command line (any characters typed after the file name) as stored in the DOS command buffer in memory. In this way, it is possible to send arguments to a program through the command string. The maximum length of a DOS command line is 20 characters.

The library routines of DOS are all included in the region of DOS preceding address 2A00(hex). For Release 5 dual-density DOS, command and I/O processing are handled by code from 2A00(hex)-2CFF(hex). It is possible to GO to a file with a GO-address in the range 2A00(hex)-2CFF(hex). However, upon return or re-entry to the DOS, the DOS routines in that region will have been overwritten, and no command processing will be possible. Instead, the Release 5 dual-density DOS will print the message:

RE-BOOT

and await an input character from the console terminal. After a system softwae diskette is loaded and a character is typed, the DOS will be re-booted from the disk.

JP <hex RAM address>

This command will cause the computer to jump to the specified

RAM address. It provides a way of executing programs which exist in the address space of the computer. Do not confuse this command with the GO command. However, like the GO command, JP sets the HL register pair to point to the remainder of the command string.

LF <file name> <hex RAM address>
SF <file name> <hex RAM address>

These commands may be used to load or save a disk file to or from RAM. The entire contents of the file will be read to or written from the area starting with the specified RAM address.

RD <disk address> <hex RAM address> <# of blocks> <optional density>
WR <disk address> <hex RAM address> <# of blocks> <optional density>

These commands may be used to read or write a specified drive directly to or from RAM. The WR and RD commands should be used with great care, as typing errors can have catastrophic effects. The disk address may optionally be followed by ",1", ",2", ",3" or ",4" to indicate a particular drive. Otherwise, drive #1 is assumed. Drive #4 may not be specified in single-density systems. The amount of data to transfer is specified as 256-byte file blocks. The optional density specification is a single letter, either "S" for single-density or "D" for double-density. If the density specification is omitted, double-density is assumed. (The single-density DOS, however, will ignore this argument.) Note that a method of copying one diskette to another in a single drive system would involve repeated use of the RD and WR commands.

IN <optional drive number> <optional density>

This command is used to initialize each new diskette to be used in the system. The IN command writes each block on the specified drive with ASCII blank characters (20 hex). The optional density argument, "S" or "D", may be used to specify whether the diskette should be initialized in single or double density format. If this argument is omitted, the diskette will be initialized to double-density. (The single-density version of the DOS will ignore the density specification.) This procedure initializes the directory and also guarantees that no "hard disk error" can result from access to an uninitialized file block. The IN command takes about 15 seconds. Needless to say, one should make sure that the proper diskette is loaded before issuing the IN command. Note that an initialized diskette does not contain a copy of the DOS. The IN command does not require any buffer area outside of the DOS memory area.

DISK SYSTEM START-UP

After power-on, or when it is desired to re-start the disk system, the 8080 or Z80 computer must be forced to begin execution at the PROM bootstrap program starting address (E800 hex in the standard version). The PROM bootstrap program will read a sector from drive #1, disk address 4 into RAM at the DOS starting address (2000 hex in the standard version). After reading in the sector, the bootstrap will branch to the DOS starting address. The program in the first block of the DOS will proceed to read in the remaining sectors of the DOS from disk starting at address 5. Then the DOS will print the prompt character (* or +) and await a command from the terminal.

Once the DOS has been started, it is no longer necessary to leave the diskette in drive #1. The DOS is fully resident in RAM, and makes no disk accesses unless asked to do so. Furthermore, the DOS does not maintain any copies of the diskette file directory in RAM between commands. Thus it is possible, for example, to obtain listings of the file directories of several diskettes by inserting them one at a time and then issuing the LI command. Also, it is possible to copy one diskette to another in a single drive system by repeatedly exchanging diskettes and doing the appropriate sequence of RD and WR commands or LF and SF commands.

DISK ERRORS

Most disk operations are tried 10 times by the DOS before reporting failure. Upon failure, an error message of the following form is printed on the console terminal:

```
DISK ERROR TYPE: x DRIVE: y SECTOR: zzz
```

where x=the error type,
y=the drive number on which the error occurred, and
zzz=the disk address at which the error occurred.

The error types have the following meanings:

1 SYNC BYTE NOT FOUND: Indicates badly written data on the diskette, or a diskette not properly loaded into the drive, or an attempt to read an uninitialized diskette.

2 CRC COMPARE ERROR: Indicates badly written data.

3 VERIFY COMPARE ERROR: Indicates data on disk does not compare with RAM in a verify operation.

4 NO INDEX PULSE: Indicates wrong type of diskette or badly loaded diskette.

5 DENSITY MISMATCH: Indicates single-density data found where double-density data was expected or visa vera.

6 WRITE PROTECT: Indicates a write operation was attempted to a write protected diskette.

If the DOS prints a question-mark(?) in response to a command, this indicates illegal form for the command or an illegal argument value.

DOS LIBRARY ROUTINES

This section describes how user machine language software may interface to the DOS for the accessing of disk files.

The DOS ENTRY POINTS AND FLAGS section shows the entry points for each of the routines to be described here. The exact interfacing requirements are described in that section. The DOS uses the stack pointer existent at call time, and some of the DOS library routines may require as much as 30 bytes of stack storage. Note that the DOS may be re-entered without using the bootstrap PROM. Now follows a discussion of each library routine.

DLOOK

This routine searches for a specified file name in the directory of the indicated disk drive. If the specified name begins with a blank, then an "empty" file directory entry is looked up. On failure to find the requested entry, HL is set to the value of the first free disk address on the indicated drive following the last file on the diskette. The file name must be in the correct syntax.

On success, HL contains a pointer into a buffer in DOS RAM that has a copy of the sought entry. The pointer addresses the first byte following the symbolic name (i.e., byte 8). Also, on return, the ACC specifies the disk drive which was determined from the name passed as argument.

DWRIT

This routine is used to write back to diskette an updated file directory entry which was previously found using DLOOK. No disk activity may occur between the DLOOK and the DWRIT call.

DCOM

This routine may be used to issue an arbitrary disk read or write command. On a read request, DCOM will try 10 times for a successful read before giving up and branching to HDERR. DCOM will fail return if the supplied arguments are out of bounds. However, great care should be used to avoid calling DCOM with incorrect arguments.

DOS

This is an entry point to the DOS command processor. It can be used to return control to a loaded DOS without requiring a PROM bootstrap load.

DOSERR

When a control-C is typed at the console terminal during a diskette directory listing, or when DOS is passed a file name which is syntactically incorrect, DOS branches to the JMP instruction stored at this location. If left unmodified, the DOSERR JMP transfers control back to a DOS error-handling routine. Modifying the address contained in

this JMP instruction will allow a user's application program to retain control under the above-named error conditions.

HDERR

HDERR branches to DOS code that prints an error message and then enters the DOS command processor. DOS branches to HDERR whenever a read attempt fails despite 10 retries. For your software to retain control in the event of a hard disk error, it must modify the address of the HDERR JMP instruction (e.g., LXI H,ADDR; SHLD HDERR+1). The stack is set to the stack pointer value before the call to DCOM. HL is set to the disk address at which the error was discovered. [Note: Software for dealing with hard disk errors is notoriously difficult. It is suggested that due to the expected low frequency of hard disk errors, for most applications the existing HDERR action will be sufficient. Hard disk errors will result primarily from careless use (e.g. forgetting to initialize a diskette, or from removing a diskette while writing is in progress). Hard disk errors can also result from power failure during writing, or from a hardware system failure.]

LIST

This routine will list the file directory of the specified drive. The listing format will be exactly the same as the listing format obtained with the DOS LI command.

OFTEN

This routine is called at least once every 40 milliseconds when DCOM has been called to perform disk operations. In the delivered copy of DOS, this routine simply does a RET. However, OFTEN may be personalized to a routine to poll for input/output requests or to enable and disable interrupts. The OFTEN routine may execute as long as is needed, and disk activity will continue when the OFTEN routine returns. OFTEN must preserve all registers except the accumulator and may only use two bytes of stack space. Note that OFTEN will be called at bootstrap load time, even before the 2900 personalization block is loaded.

Note: Here is a procedure for creating a new file using the above routines: First use DLOOK to search for the desired new name - if DLOOK succeeds then a file of that name aready exists and should not be created. On failure, HL will have the disk address which should be used as the starting address of the new file. Next, use DLOOK to find an empty directory entry by looking up a blank name. If this call to DLOOK fails, then the directory fails. On success, use the pointer in HL to copy the new file name into the directory entry, and copy in the disk address and length and type information. Finally, call DWRIT to copy the new directory entry back to the disk.

ADDITIONAL DOS PERSONALIZATION

The primary type of personalization that can be done to the DOS is the insertion of input/output routines that allow communication of the DOS and other system software with a particular hardware configuration. Input/output routine personalization is described in detail in the GETTING STARTED section of this manual. There are a number of other types of personalization that can be done to the DOS that are described in this section.

READ AFTER WRITE CHECK

If the read after write check option is turned on, then a read and verify operation is performed after every disk write operation which checks that the data written on the disk by the write operation matches what is in RAM. With this option turned on, write operations will be slower, but read operations will be the same speed. It is strongly recommended that the read after write option be turned on unless the application requires great speed of disk access. The read after write option is turned on if the byte value at address 202B(hex) of the standard version of DOS is non-zero and turned off if zero.

PAGE SIZE

The output of some devices, such as CRT's and video displays, can only display a fixed size page of information at one time. If the page size option is enabled, then the file directory listing which is output by the LI command or the LIST library routine will stop after a page of information has been output and will not display the next page until the user indicates he wishes to proceed by typing the return key. The page size option only affects the operation of the console terminal (device #0). If the byte value at address 2033(hex) of the standard version of DOS is zero, then the page size option is turned off and output will be continuous without stopping. If the value is non-zero, then the value is the number of lines on a page and the output will stop after that many less one lines of output have been displayed. The last line on the page will request the user to type return to continue. Initially, the page size option is on and set for a page size of 24 lines.

AUTOMATIC START

If the automatic start option is turned on, then a single command which is stored in the DOS input buffer is automatically executed immediately after a DOS bootstrap operation. This feature, for example, allows for the automatic loading and running of a program such as BASIC upon system start-up. Initially, the automatic start option is turned off. To turn on the option, the byte value at address 2030(hex) of the standard version of DOS should be set to zero. Initially, the value is one. Additionally, the input buffer must be setup to contain the

command which should be automatically executed. The two addresses 2031 and 2032(hex) contain the low order and high order byte, respectively, of the address of the input buffer within the DOS. The input buffer should be loaded with the ASCII values of the successive characters of the desired command. The last character of the loaded command must be a return (ØD hex).

EXAMPLE PERSONALIZATION

The following listing shows an example procedure which will modify the version of DOS on a diskette so that the read after write option is turned on, the page size option is turned on and the page size is set to 24 lines, and the automatic start option is turned on and the automatic start command is set to be "GO BASIC".

Load the diskette to be modified in drive #1

```
+LF DOS 4000        Load DOS into RAM
+GO M2DØØ           Load and run the Monitor
>FM 4Ø2B 1          Turn on read after write option
>FM 4033 24T        Set page size to 24 lines
>FM 4030 Ø          Turn on automatic start option
>DH 4031,2          Determine input buffer address
402D: 31 27         Address is 2731, for example
>FM 4731 "G"        Load input buffer with "GO BASIC"
>FM 4732 "O"
>FM 4733 " "
>FM 4734 "B"
>FM 4735 "A"
>FM 4736 "S"
>FM 4737 "I"
>FM 4738 "C"
>FM 4739 D          Put return code at end of command
>OS                 Return to DOS
+SF DOS 4000        Save modified DOS back on the diskette
```

```
0000            *
0000            *NORTH STAR DISK OPERATING SYSTEM
0000            *
0000                  ORG 2000H          STANDARD VERSION ORIGIN VALU
2000                  DS 7               THESE CELLS ARE RESERVED
2007            *
2007            *THE OFTEN ROUTINE IS CALLED FREQUENTLY DURING USE O
2007            *BC, DE, AND HL MUST BE PRESERVED BY OFTEN.
2007            *ONLY TWO STACK BYTES ARE AVAILABLE.
2007 C9         OFTEN RET          CHANGE TO JMP INSTRUCTION
2008                  DS 2          IF ADDING YOUR OWN OFTEN ROUTINE
200A            *
200A            *THIS NEXT ENTRY IS USED BY THE BOOT PROM TO ENTER T
200A C30000     START JMP 0        0 IS NOT THE REAL ADDRESS
200D            *
200D            *THIS IS THE CHARACTER OUTPUT ROUTINE
200D            *THE CHARACTER TO BE OUTPUT MUST BE IN THE B REGISTE
200D            *DEVICE NUMBER MAY BE SUPPLIED IN ACC, IF DESIRED.
200D            *ON RETURN THE CHARACTER MUST ALSO BE IN THE ACC.
200D            *ONLY THE ACC AND FLAGS MAY B MODIFIED
200D C30D20     COUT  JMP COUT     YOUR ROUTINE MUST DO A RET
2010            *
2010            *THIS IS THE CHARACTER INPUT ROUTINE.
2010            *DEVICE NUMBER MAY BE SUPPLIED IN ACC, IF DESIRED.
2010            *THE 7-BIT ASCII CODE MUST BE RETURNED IN THE ACC.
2010            *ONLY THE ACC AND FLAGS MAY BE MODIFIED.
2010 C31020     CIN   JMP CIN      YOUR ROUTINE MUST DO A RET
2013            *
2013            *THIS IS THE TERMINAL INITIALIZATION ROUTINE
2013            *ALL REGISTERS MAY BE USED.
2013            *IF NOT NEEDED, MERELY PATCH IN A RET.
2013 C31320     TINIT JMP TINIT
2016            *
2016            *THIS ROUTINE DETECTS A CONTROL-C
2016            *IF Z IS SET ON RETURN, THAT MEANS A CONTROL-C WAS
2016            *OTHERWISE, IF NO CHARACTER WAS TYPED OR A CHARACTE
2016            * THAN CONTROL-C WAS TYPED, Z MUST NOT BE SET.
2016            *CONTC SHOULD RETURN IMMEDIATELY IF NO CHAR WAS TYP
2016            * NOT WAIT FOR A CHARACTER AND THEN RETURN.
2016            *ALL REGISTERS MAY BE USED.
2016 C31620     CONTC JMP CONTC
2019            *
```

DOS ENTRY POINTS AND FLAGS (Continued)

```
2019                *DOS LIBRARY ROUTINE ENTRY POINTS, ETC.
2019                *
2019                *THIS ADDRESS IS BRANCHED TO ON HARD DISK ERRORS
2019 C30000         HDERR JMP 0         0 IS NOT THE REAL ADDRESS
201C                *
201C                *THIS IS THE FILE DIRECTORY LOOKUP ROUTINE
201C                *ACC MUST CONTAIN THE DEFAULT UNIT NUMBER (NORMALLY 1)
201C                *HL=POINTER TO LEGAL FILE NAME IN RAM, WITH OPTIONAL DRIVE
201C                * SPECIFICATION FOLLOWED BY EITHER A BLANK OR CARRIAGE RETURN.
201C                *UNIT NUMBER DETERMINED FROM NAME IS ALWAYS RETURNED IN ACC.
201C                *FAILURE IF CARRY SET. ON FAILURE, HL=FIRST FREE DISK ADDRESS
201C                *ON SUCCESS, HL HAS A POINTER TO THE EIGHT BYTE OF A COPY
201C                *OF THE DOS ENTRY IN RAM
201C C30000         DLOOK JMP 0         0 IS NOT THE REAL ADDRESS
201F                *
201F                *THIS ROUTINE WILL WRITE A DIRECTORY ENTRY BACK TO DISK
201F                *NO ARGS ARE NEEDED. MUST FOLLOW DLOOK.
201F C30000         DWRIT JMP 0         0 IS NOT THE REAL ADDRESS
2022                *
2022                *THIS ROUTINE MAY BE USED TO ISSUE A DISK COMMAND
2022                *ACC=NUMBER OF BLOCKS
2022                *B=COMMAND (0=WRITE, 1=READ, 2=VERIFY, -1=SING INIT, -2=DBL INIT)
2022                *C=UNIT NUMBER, BIT 7=DOUBLE DENSITY BIT
2022                *DE=STARTING RAM ADDRESS, HL=STARTING DISK ADDRESS
2022                *RETURN WITH CARRY SET MEANS ARGUMENTS WERE ILLEGAL
2022 C30000         DCOM  JMP 0         0 IS NOT THE REAL ADDRESS
2025                *
2025                *THIS ROUTINE MAY BE USED TO LIST A FILE DIRECTORY
2025                *ACC=DISK UNIT, L=OUTPUT DEVICE NUMBER FOR LISTING
2025 C30000         LIST  JMP 0         0 IS NOT THE REAL ADDRESS
2028                *
2028                *THIS ADDRESS IS AN ENTRY POINT TO THE LOADED DOS
2028                *ENTRY HERE WILL RESET THE STACK PTR, AND NOT CALL TINIT
2028 C30000         DOS   JMP 0         0 IS NOT THE REAL ADDRESS
202B                *
202B                *THIS NEXT BYTE IS A FLAG USED BY DOS.
202B                *IF 0, THEN READ-AFTER-WRITE CHECK IS NOT DONE,
202B                *IF 1, THEN READ-AFTER-WRITE CHECK IS DONE.
202B 00             RWCHK DB 0
202C                *
202C                *THIS ADDRESS BRANCHED TO ON CONTROL-C DURING LIST OR
202C                *FILE NAME ERROR DURING DLOOK
202C C30000         DOSERR JMP 0          NOT REALLY 0
202F                *
202F                *THIS BYTE IS SET TO DENSITY AFTER DLOOK CALLS
202F                *00H IF SINGLE DENSITY, 80H IF DOUBLE DENSITY
202F                DEN DS 1
2030                *
2030                *AUTO START FLAG. NORMALLY 1 - SET TO 0 FOR TURNKEY STARTUP
2030 01             AUTOS DB 1
2031                *
2031                *NEXT TWO BYTES IDENTIFY THE LOCATION OF THE DOS INPUT BUFFER
2031 0000              DW 0             NOT REALLY 0
2033                *
2033                *NEXT BYTE SPECIFIES VIDEO TERMINAL LINE COUNT. IF 0, THEN
2033                *NO PAGING OF THE LIST COMMAND WILL BE DONE
2033 18             PAGES DB 24         INITIALIZED FOR 24 LINE TERMINAL
2034                *
```

UTILITIES

There are four operations which may be considered as part of the DOS but are actually implemented as GO files. The operations, and their corresponding GO file names are:

DT - Disk Test.
CF - Copy File.
CD - Copy Disk.
CO - Compact disk and convert to double-density.

Complete descriptions of the utilities follow. Some of the arguments to the utilities can be listed on the command line where "GO" is typed. For example

```
+GO DT 1
```

may be typed to the DOS. This tells the DT utility which drive is to be tested. Any arguments which you do not supply to the utility on the GO command line are explicitly requested by the utility.

The origin in memory of each of the utilities lies just after the end of DOS (2A00H in single-density systems and 2D00H in double-density systems). Each of the utilities requires a 5K buffer area (2.5K in single-density systems). The amount of RAM required by a utility may be computed by adding the buffer size to the size of the utility on diskette. Because the utilities load at the same address as the standard version of BASIC and many other applications programs, you should be careful that no programs or data be overwritten and therefore lost as a result of using a utility.

You may wish to use a utility to operate on a diskette different than the diskette that holds the utility program. In this case, you must change diskettes after the utility has been loaded into RAM. Each of the utilities allows a different diskette to be loaded before actually beginning its operation. Diskettes can be switched any time after the utility makes its first request for input. Do not answer that request until the switch, if any, has been made!

In the following expanded descriptions of the utilities, any references to double-density capability refer only to versions of the utilities for use on double-density systems.

Typical user-computer interaction at the terminal is given as EXAMPLES for each of the utilities. In these examples, note that the DOS prompt given is a plus-sign (+). However, single-density versions of the DOS generate an asterisk (*) as prompt. In examples, the symbol <CR> comes immediately after the user's responses to indicate that a line of user input must always be terminated by striking the RETURN key.

DT - Disk Test.

The Disk Test utility tests the specified drive and the diskette loaded in that drive. The following cycle is continuously repeated:

a) The entire diskette is written with data, starting at sector Ø. An incrementing pattern is used. If the read after write check is enabled (see DOS section ADDITIONAL DOS-PERSONALIZATION), then each track is verified immediately after it is written.

b) The data on the entire diskette is verified, starting at sector Ø. If any sector cannot be read or contains data different than what was written, an error messgae is printed on the console terminal and the test stops.

c) If no errors have been detected by this point, the message

   PASS COMPLETED.

   is printed on the console terminal.

To terminate a disk test, type control-C. A diskette used for a disk test does not emerge from the test containing the information which was previously on it. Also, a diskette which was used for a disk test must be initialized before it is subsequently used for data storage.

EXAMPLES

```
+GO DT<CR>
DRIVE NUMBER: 2<CR>
SINGLE(S) OR DOUBLE(D) DENSITY TEST? D<CR>
LOAD DISKETTE AND PRESS RETURN TO BEGIN TEST.<CR>
PASS COMPLETE.
PASS COMPLETE.
PASS COMPLETE.
CONTROL-C STOP        User types control-C here.
+


+GO DT 2 D<CR>
LOAD DISKETTE AND PRESS RETURN TO BEGIN TEST.<CR>
DISK ERROR TYPE 3 DRIVE 2 SECTOR 352
+
```

CF - Copy File.

The Copy File utility copies the contents and type information from a source file to a destination file. The destination file may be a file which already exists, but if it does not, it is created automatically. If the destination file already exists it must be at least as large as the source file (in 256-byte file blocks). Whether the destination file exists or not, CF asks if the destination file should be written in double or single density. The source and destination files may be on different diskettes loaded on different drives, or they may be on the same diskette.

If any sectors in a source file are recorded in a density different than the density specified in the directory entry, the CF utility treats those sectors as sectors full of blanks at the specified density. No change is made to the source file, however.

Note that versions of the CF utility delivered for single-density systems only provide single-density operation.

EXAMPLES

```
+GO CF<CR>
FROM FILE: TEST<CR>
TO FILE: PROGRAM,2<CR>
EXISTING FILE. SINGLE(S) OR DOUBLE(D) DENSITY? D<CR>
COPY COMPLETED.
+


+GO CF ABC ABC1<CR>
NEW FILE.  SINGLE(S) OR DOUBLE(D) DENSITY? S<CR>
COPY COMPLETED.
+
```

CD - Copy Disk.

The Copy Disk utility copies the entire contents of a diskette loaded on one specified drive to a diskette loaded in another specified drive. The source diskette may contain single-density information, double-density information, or a combination of the two. After the copy is completed, the destination diskette will contain all the same information as the source diskette, and each sector will be recorded in the same density as the source. If any information on the source diskette is impossible to read, the copy terminates. The copy operation can be retried after the bad sector has been rewritten.

EXAMPLES

```
+GO CD<CR>
COPY FROM DRIVE: 1<CR>
TO DRIVE: 2<CR>
LOAD DISKETTES AND PRESS RETURN TO BEGIN COPY.<CR>
COPY COMPLETED.
+
```

```
+GO CD 2 3<CR>
LOAD DISKETTES AND PRESS RETURN TO BEGIN COPY.<CR>
COPY COMPLETED.
+
```

CO - Compact.

The Compact utility is used to "compact" the file space on a diskette. Any unused disk space between existing files is eliminated by moving the files toward track 0. Thus, the CO utility can be used to reclaim disk space after files have been deleted or shortened, or in case files were created in such a way as to leave gaps of disk space between them.

The CO utility also provides a second, optional function which converts a diskette to double-density format. That is, as a result of running CO, the diskette file directory will be recorded in double-density, and all files that were previously single-density files will become double-density files. Each pair of single-density file blocks (256-bytes per block) is stored in one double-density sector (512 bytes).

Before actually beginning to move files on the diskette, CO checks the file directory for any "overlapping" files. Overlapping files are any files which include at least one sector in common. Overlapping files can only be created when the optional <disk address> argument is used with the DOS CR command, or by applications programs which create such files. If any overlapping files are discovered by the CO check, the file names are printed on the console terminal and the user is given the opportunity to abort the compaction. If overlapping files exist, the compaction may yield unpredictable results. (NOTE: The special case of a file with disk address beginning at 0 is ignored by this check, and by the compaction process.)

A compaction can take from 1 to 30 seconds.

EXAMPLES

```
+GO CO<CR>
LOAD DISKETTE AND SPECIFY DRIVE #: 1<CR>
CONVERT TO DOUBLE DENSITY? Y<CR>
COMPACTION COMPLETED.
+


+GO CO 3<CR>
CONVERT TO DOUBLE DENSITY? Y<CR>
THE FOLLOWING FILES HAVE CONFLICTS
DATA1
TEST123
PROCEED WITH COMPACT IN PRESENCE OF CONFLICTS? N<CR>
+
```

North Star Monitor

Version 2

by Thos Sumner

INTRODUCTION

The North Star Monitor is a program which provides the user with certain maintenance and debugging functions which would normally be provided in a limited way on systems which include a control panel. The Monitor is intended to be used in conjunction with the North Star Disk Operating System (DOS). No license is extended for use of the Monitor in systems without a North Star disk controller board.

Commands to the Monitor are entered via the console using a format consistent with the DOS commands. The console is defined to be the terminal with which the DOS normally communicates - communication is done using the DOS I/O routines. When the Monitor is in COMMAND MODE, i.e., is ready to accept a command, it will print a > at the beginning of a line on the console. Command editing facilities compatible with the North Star BASIC editing features are included in the Monitor.

The following list summarizes the commands available:

CM - Compare memory block contents
FM - Fill memory block
MM - Move memory block contents
SM - Search memory block
TM - Test memory block
DH - Display memory hexadecimal
DA - Display memory with ASCII interpretation
DS - Display memory and substitute values
JP - Jump to program
OS - Return control to the DOS
IL - Perform initial load from bootstrap PROM
OD - Assign output device number for the Monitor
ID - Assign input device number for the Monitor

A detailed description of each command appears in a later section below. All printed output from the Monitor is formatted to fit into sixty-four character lines.

COMMAND FORMAT

No action is taken on any command until the command is fully entered and all editing functions are complete (i.e. when a carriage return is typed). Each command is fully checked for syntax errors before any part of the command is performed. Thus, a command aborted because of illegal syntax will have no effect.

The command syntax is quite simple. Each command appears on a single line and begins with a two letter mnemonic name which identifies the command. Upper case (capital) letters must be used for commands. The command name is followed by any required parameters which are separated from the command name and each other by spaces (blanks). Spaces may not appear within a parameter (except within quotation marks as described later). The command is terminated by a carriage return.

Commands may take no parameters or may take one or more of several types of parameters. The parameters required for Monitor commands have the following forms:

NUMBER — A NUMBER may be any of the following:

1. A hexadecimal number specified by a sequence of one or more hexadecimal digits.
2. A hexadecimal number specified by a sequence of one or more hexadecimal digits followed by the letter H.
3. A decimal number specified by a sequence of one or more decimal digits followed by the letter T.

ADDRESS — An ADDRESS is a NUMBER in the range from Ø though 65535T (Ø through FFFFH).

BLOCK — A BLOCK is a contiguous region of memory specified by one of the following:

1. An ADDRESS alone specifies a one byte block at the given memory address.
2. Two ADDRESSES separated by a hyphen (-) specify a block whose first byte is at the first address given and whose last byte is at the second address given (e.g., 2000-3FFF). The second address may not be less than the first.
3. An ADDRESS followed by a NUMBER separated by a comma specifies a block whose first byte is at the ADDRESS given and whose length is given by the NUMBER (e.g., 2000,1000). The address of

the last byte (ADDRESS + NUMBER - 1) may not exceed 65535T (FFFFH).

BYTE VALUE — A BYTE VALUE is a parameter whose value fills a single byte. It may be specified by either of two forms:

1. A NUMBER in the range 0 though 255T (0 though FFH), or

2. A non-control character in quotation marks (e.g., "A"). A non-control character is any character entered without using the control key or other function keys such as carriage return, line feed, tab, etc.

DEVICE NUMBER — A DEVICE NUMBER is a cross-hatch (#), followed by a digit from 0 to 7, corresponding to one of the 8 possible I/O devices which have been interfaced to your DOS through the DOS I/O routines. Note that a DEVICE NUMBER may be given immediately after the code for any Monitor command which produces output. As a result, the output produced by the execution of that command will be printed on the specified device. For example, to send a hexadecimal dump of memory from 2000H-29FFH to system output device 4:

```
>DH #4 2000-29FF
```

Also note that only the output for the given command will be re-directed. After the command has been executed, output resumes on the regular Monitor output device (as set by the OD command).

This section gives a description and example for each Monitor command. An expanded example of the use of the Monitor follows in a later section. Note that the typing of commands can be corrected using all the line editor features described in the LINE EDITOR chapter of the BASIC section of this manual.

CM <block> <address>

Compare the memory area defined by BLOCK with the area of the same length starting at ADDRESS and print the address and contents of all corresponding bytes which are not identical. For example, to compare the contents of the seven bytes starting at 3400H with the seven bytes starting at E385H:
>CM 3400,7 E385

FM <block> <byte value>

Fill each byte of the memory area defined by BLOCK with the value specified by the BYTE VALUE. For example, to fill the block of memory from 4000H to 5000H with the ASCII blank character:
>FM 4000-5000 " "

MM <block> <address>

Move the contents of the memory area defined by BLOCK to the corresponding positions in the area of the same size beginning at ADDRESS. Overlapping areas of memory are allowed and the moves are performed correctly. For example, to move the contents of the one hundred (decimal) byte block starting at address 9000 (decimal) into the area starting at F000H:
>MM 9000T,100T F000

SM <block> <list of byte values>

Search the area of memory defined by BLOCK and print the addresses of all occurrences of the sequence of bytes specified in the LIST OF BYTE VALUES. The list must contain at least one byte value. If it contains more than one, commas must separate the various byte values from one another. For example, to find all bytes in the standard DOS area whose value is one:
>SM 2000-29FF 1

To find all occurrences of the sequence "MINE" in the block 4000H-5000H:
>SM 4000-5000 "M","I","N","E"

It is not necessary for the sequence of bytes defined in a LIST to be wholly contained within the specified BLOCK. SM will report the locations of all occurrences of the sequence as long as the first character in each occurrence lies within

the BLOCK.

TM <block> <number>

Test the memory area defined by BLOCK. The NUMBER parameter specifies the delay (in seconds) between a write phase of the test and the subsequent verify phase. The delay allows testing memory for failures resulting from gradual decay of data. The number may have a value from 0 to 255T (0 to FFH). For example, to test the 8K memory located at 6000H without delay between the write phase and the verify phase:
>TM 6000H,2000 0

The test operates as follows: Values are written into every byte in the block (write phase) and then the values are checked (verify phase). In a complete pass, 256 write and verify phases occur, so that every possible value is tested in every byte. The test is designed to catch addressing errors as well as data bit errors. The test requires about 15 seconds per thousand bytes of memory for a complete pass (plus 256 times the delay value). The test will run continuously and can only be halted by typing a control-C. Note that even running the test for a few seconds will perform a quick check on the entire block. After each complete pass, the TM command will print PASS COMPLETE on the terminal. For every memory error detected, the address of the problem is printed in hexadecimal followed by the value stored at that address and the value found at that address.

DH <block>

Display the contents of the memory area defined by BLOCK on the console in a format of two hexadecimal digits per byte with sixteen bytes on each line. For example, to print the contents of memory from 12000 (decimal) to 12099 (decimal) in hexadecimal:
>DH 12000T,100T

DA <block>

Display the contents of the memory area defined by BLOCK in the format of the DH command but with additional lines showing the ASCII character represented by the low order seven bits of each byte positioned immediately under the corresponding hexadecimal digits. A control code is printed as a blank and each character is preceded by a minus-sign if the high order bit of the byte is a one. For example, to print the contents of memory from 12000 (decimal) to 12099 (decimal) in both hexadecimal and as ASCII characters:
>DA 12000T-12099T

DS <address>

Display the current contents of the memory area starting at ADDRESS in hexadecimal, one byte at a time, and allow the option of substituting a new value for each byte. After each byte is displayed, if it is desired to modify the contents of the displayed cell, then a new hexadecimal value in the range Ø to FF may be entered. If it is desired to leave the byte unchanged, then type one of the following terminators immediately. Whether or not a new value is entered, the terminator character typed determines the subsequent action:

1) Blank. If a blank is typed then the next byte will be displayed for possible replacement.

2) Comma. If a comma is typed, then the printing of the next byte is suppressed, and subsequent typing will either replace or leave unmodified that next cell.

3) Carriage return. Typing a carriage return finishes the command and returns to command mode.

Note that the replacement occurs as soon as one of the terminators is typed, so that the line editor may not be used as in other commands. However, a typing error in specifying a new value may be corrected with any of the backspace commands before the terminator is typed. For example, to replace the contents (currently 64H) of the byte at 3233H with a zero:

```
>DS 3233
3233 64=Ø
```

JP <address>

Cause the computer control to jump to the specified ADDRESS. The Monitor performs a CALL instruction to that address, so that if the program executed at that address later performs a RET and has preserved the stack pointer and not destroyed any of the memory of the Monitor, then control will return to the Monitor for another command. For example, to execute the normal continue location for BASIC:

```
>JP 2AØ4
```

Note that when the JP command is executed, the HL register pair is set to point to the remainder of the command which was typed in (any characters typed after the ADDRESS). This makes it possible to send parameters through the JP command to the program being executed.

OS

Jump to the standard re-entry address of the DOS. This command requires no parameters.

IL

Jump to the bootstrap startup PROM to perform an initial load of the DOS from diskette. This command requires no parameters.

OD <device number>
ID <device number>

Sets the Monitor's output or input device number, respectively, to the specified DEVICE NUMBER. During Monitor output and input, these numbers are sent to the DOS I/O routines, and may be used by them to determine routing of output and input to and from the appropriate system I/O devices. The value supplied to OD or ID must be in the range of Ø to 7. Both the input and output device numbers are set to zero when the Monitor is entered, corresponding to I/O from the console terminal. When an OD or ID command is executed, the new device number remains in effect for all output or input, respectively, produced by the Monitor until another OD or ID command is typed. (Note that these device numbers will also remain in effect upon re-entry to the Monitor at its initial entry point.) For example, to cause output to appear at output device 1:

```
>OD #1
```

To cause input to be accepted from device 6:

```
>ID #6
```

INTERRUPTING THE MONITOR

Some commands may compute for long periods of time and/or produce substantial amounts of output. The CM (compare memory), SM (search memory), DH (display hexadecimal), DA (display ASCII) and TM (test memory) commands may be interrupted by typing a control-C on the console. This will interrupt the Monitor to allow a new command to be entered.

HARDWARE REQUIREMENTS

Any system operating the North Star DOS and BASIC will execute the Monitor. However, because of its applications, the Monitor is supplied in versions assembled at four different locations. Each version of the Monitor requires 2048 (800H) bytes and has its primary entry point at its origin. The standard versions are:

File name	Origin	Description
M0000	0000H	For systems with memory at origin zero, this version may be in memory along with both DOS and BASIC. It may be re-entered by a simple processor reset.
M2A00	2A00H	(single-density systems only)
M2D00	2D00H	(double-density systems only) This version uses the memory area beginning immediately after the standard DOS. Although this version can not be loaded at the same time as BASIC, it will run on any system which can run the standard version of BASIC.
M6700	6700H	This version is useful for modifying BASIC while it is in RAM.
MF400	F400H	This version is useful for systems which have memory in the F block and which can not conveniently use the address space at zero for the Monitor.

A special fifth version of the Monitor is also supplied:

M5700	5700H	This version of the Monitor is designed primarily for use in testing the RAM memory which normally contains the DOS and Monitor programs. This version of the Monitor is 900H bytes long, and the last 100H bytes contain the standard HORIZON terminal input/ouput routines. The origin of this version was chosen so that it will reside at the end of a 16K board origined at 2000H.

PERSONALIZING THE MONITOR

The Monitor begins with a series of jump instructions through which it is entered and through which it links to the DOS input/output routines*. If the Monitor is not to be used with a standard-origin DOS, then the appropriate address changes must be made to these jump instructions. Since the Monitor is supplied in several versions, note that the following discussion applies to the version with origin at zero.

Address	Instruction	Purpose
0000H	JMP INIT	This is the initial entry point to the Monitor
0003H	JMP 200DH	Jump to the DOS COUT routine
0006H	JMP 2010H	Jump to the DOS CIN routine
0009H	JMP 2028H	Jump to the DOS re-entry point
000CH	JMP 2016H	Jump to the DOS CONTC routine
000FH	JMP E800H	Jump to the PROM bootstrap address

Consult the GETTING STARTED section for details of the required routines.

The value at address 0012H is the ASCII code for the character to be echoed when one of the following line editting control characters is typed: control-Q, underline, left-arrow, rubout, or delete. Initially, this value is set to the ASCII code for underline. This value is analogous to the byte discussed in the PERSONALIZING BASIC chapter of the BASIC section of this manual.

Provision has been made for adding as many as four user-defined commands to the Monitor. Approximately 240H bytes after the origin of the Monitor there is a block of seventeen (11H) bytes of zeroes which lies at the end of the Monitor command table. This provides space for adding up to four entries of four bytes each to define user commands and allows for the zero byte which always terminates the command table.

Each command definition requires four bytes: the first pair of bytes contains the two-letter command name. The second pair of bytes contains the address (low-order byte first) of the routine to be called to execute the command. The program to implement the command should preserve the stack pointer and re-enter the Monitor with a RET instruction. No more than 30 bytes should be pushed onto the Monitor stack during execution of the command.

* Except for the M5700 version of the Monitor which has links to the input/output routines in its last 256-byte block (5F00-5FFF). If non-standard input/output routines are required for your hardware configuration, then new routines which correspond to the routines in the DOS at 2900-29FF must be loaded in the region 5F00-5FFF, and the M5700 jump table must be fixed accordingly.

EXAMPLE

This section will show how the Monitor can be used to modify itself for the addition of a new command. The command will have the name AA, and the code for the command has been generated onto a disk file named CODE, assembled for execution beginning at 800H. The new version of the Monitor will be a version that runs at 0.

+LF M0000 4000	Load the standard version of the origin zero Monitor to RAM.
+LF CODE 4800	Load the code for the new Monitor command immediately after the copy of the Monitor in RAM.
+GO M2D00	Load and execute the standard Monitor.
>SM 4000,800 0	Find the zero block for command table expansion. Locate the first block of seventeen zeroes (assume for this example that the location of the first zero in the block is at address 4241H).
>FM 4241 41 >FM 4242 41 >FM 4244 08	These commands add the command name AA to the command table and set the start address of the code for the command to be 800H.
>OS	Return to the DOS.
+CR NEWMON 10	Create a new file, two blocks larger than the standard Monitor, to contain the modified version of the Monitor.
+TY NEWMON 1 0	Set the type of the new Monitor file to 1 and set the go-address to 0.
+SF NEWMON 4000	Save the expanded new Monitor to disk.
+GO NEWMON	Load and execute the new Monitor
>AA	Test the new command.

INTRODUCTION

THE NORTH STAR BASIC SYSTEM

Version 6

by Jim Merritt

ABOUT NORTH STAR BASIC

North Star BASIC was created by Dr. Charles A. Grant and Dr. Mark Greenberg of North Star Computers, Inc. This manual describes version 6, an extended disk BASIC intended for use with the North Star HORIZON computer or MICRO DISK SYSTEM. Version 6 includes many features especially designed to facilitate scientific, business, and industrial applications programming. Of special note are North Star BASIC's facilities for programmed error handling, automatic program sequencing (CHAINing), formatted output, sophisticated string handling, and machine language subroutine interface. Both single line and multiple line user-function definitions are supported, as well as multiple-dimension numeric arrays, and complete disk file handling capabilities. Data files may be accessed sequentially, randomly, or on a byte by byte basis. North Star BASIC combines all these "extras" with the usual features found in any reasonable implementation of BASIC, to yield a unique development tool which promotes the writing of powerful BASIC programs. Special design features ease the task of "converting" programs written for other BASIC systems so that they will run under North Star BASIC. BASIC is also supplied in a version which uses the North Star Hardware Floating Point Board (FPB-A). The two versions, Floating Point and Non-Floating Point, are identical in features and operation but the FPB version executes arithmetic operations faster.

The North Star Version 6 BASIC software is intended for use only with the North Star HORIZON computer or MICRO DISK SYSTEM, and no license is granted for any other use. Improved copies of Version 6, as they become available, may be obtained for a nominal charge.

HOW THIS MANUAL IS ARRANGED

This manual attempts to meet the needs of both the novice programmer, with little or no BASIC background, and the experienced BASIC programmer, who needs only know the particular characteristics of North Star BASIC.

For the expert, individual STATEMENTS, COMMANDS, and other specific language features are covered in their own brief

exposition sections. Each exposition consists of the following:

SYNTAX GUIDE: This includes one or more brief models which define the form of the STATEMENT or COMMAND within North Star's BASIC syntax.

ACTION: This tells what happens when the STATEMENT or COMMAND is used.

EXAMPLES (or EXAMPLE PROGRAMS): These show the STATEMENT or COMMAND in typical use. When the feature may take a variety of forms, an attempt has been made to provide several representative examples. Frequently, the feature is illustrated in the context of a sample program or program segment.

REMARKS: Whenever necessary, this section is included to provide further information about the feature's use.

ERROR MESSAGES: Improper formation or usage of a language feature will result in a BASIC error condition which will lead to both the termination of the program or COMMAND being processed, as well as an ERROR MESSAGE sent to you. Wherever applicable, the common ERROR MESSAGES associated with improper use of a given feature, as well as their probable causes, are given in the ERROR MESSAGES section for that feature. Note that common error messages which apply generally to all STATEMENTS and COMMANDS are described in APPENDIX 2.

SEE ALSO: Here you will find cross references to relevant manual sections, study of which may help you more fully understand a given feature.

The manual includes several appendices in the back, two of which provide thorough indexing of all topics and features in the manual. Other appendices contain charts, tables, and detailed information useful to the practicing programmer.

For the beginner, there are many DISCUSSION sections, which explain the underlying concepts and capabilities of North Star BASIC. Programming methodology and strategy are also examined in these sections. This is not to say that the DISCUSSION sections should be ignored by experienced programmers. On the contrary, experts will find much useful information in these sections.

DISCUSSION and exposition sections have been interspersed throughout the manual. Furthermore, an attempt has been made to organize the manual so that elementary material is presented first, while more advanced features and concepts

are treated later. This has been done to facilitate the beginner's likely "cover to cover" approach to manual reading. While the manual is not intended as a course in BASIC programming, a thorough front to back study of it will yield much knowledge of programming in general, and programming in North Star BASIC in particular. Those who are absolute beginners in the field are referred to the introductory computer and programming texts at local libraries, book stores, and computer retail stores. If you desire instruction on the fundamentals of programming and computers, choose one such book and use it as a primer to this manual.

Finally, for all users, APPENDIX 1 contains many sample programs which illustrate the typical integration of North Star STATEMENTs and other features and capabilities into finished software.

NorthStar™

North Star Computers, Inc.
14440 Catalina St., San Leandro, CA 94577 USA
(415) 357-8500 TWX/Telex (910) 366-7001