RELEASE 3 DOS ✓

Release 3 of Version 2 DOS has several bugs fixed, and some changes also:

A) The GA command has been removed. Whenever the TY command specifies a type 1 file, then a third argument is expected which gives the go-address. E.G., TY BASIC 1 2A00.

B) The CF command now uses the 2.5K buffer immediately following the DOS.

C) Two editing characters are now accepted during the typing of DOS commands: at-sign (@) for deleting the entire current line, and underline (_) for deleting the most recently typed character. On some terminals a left-arrow character is equivalent to the underline.

D) The DLOOK subroutine may now be used to look up an "empty" entry in the directory, for the purpose of finding an entry which may be used for a new file. If the name pointed to by HL when DLOOK is called starts with a blank, then the first all-blank entry in the indicated directory will be found, if present. Thus, to create a new file using the DOS subroutines, first look up to desired new name to be sure it doesn't already exist. This lookup will return the first "free" disk address on the sepcified diskette. Next, do a lookup for a blank entry and copy in the name and disk address and length. Finally, do a DWRIT to write out the updated entry to diskette.

E) The DOS now puts 0 into the ACC for CIN and COUT calls.

F) The LI command will now list the go-address of type 1 files.

G) The personalizing instructions for BASIC on page 10 of the DOS manual should now indicate that Release 3 BASIC is to be created by doing:

```
*CR BASIC 45
*TY BASIC 1 2A00
```

RELEASE 3
NORTH STAR VERSION 6 BASIC

Release 3 BASIC fixes all reported bugs in Release 2 BASIC, and also makes several additions. Release 3 is a superset of Release 2 - all programs which run under Release 2 should run under Release 3. Note that the size of BASIC has increased - now a 45 block file is required.

A) Multiple input/output devices. Now PRINT, INPUT, LIST, and LINE may specify a "device expression" as an optional first argument preceded by a number sign, e.g.

```
PRINT #3,A,B,C
INPUT #X,"TYPE A NUMBER: ",N
LIST #2,100,500
LINE #2,64
```

Device 0 is assumed if the device expression is missing. Also, device 0 is always used for command communication in direct mode and in the DOS. LINE becomes an executable statement instead of a command, so that device line lengths can also be set up within programs as well as in direct mode.

The device value, which must be between 0 and 7 inclusive, is communicated in the ACC to the DOS CIN and COUT routines. The DOS routines can then decide which device to communicate with based on the contents of the ACC. Note that no changes are needed for the DOS I/O routines if this feature will not be used. Also note that this feature may be used to communicate string values to machine language subroutines.

B) Delete command. The DEL command has been added to allow the deletion of a block of program statements. For example,

```
DEL 100,200
```

will delete all program statements with line numbers in the range of 100 through 200 inclusive.

C) Byte access to data files. It is now possible to access a data file in BASIC at the byte level. A numeric value in the range of 0 to 255 may be written into or read from a specified byte address in a data file. These byte accesses may be performed in either sequential or random access mode. To indicate byte mode, an ampersand (&) is placed before the variable to be read into or the expression to be written out. For example,

```
10 READ #0,A,&B
```

```
20 WRITE #1,&X+1,B$
30 READ #0 %X,&T,&T1
40 WRITE #F,%X,&ASC("a")
```

D) Determining file length. The length of a data file may be determined by adding an optional variable name at the end of the OPEN statement. After the OPEN statement, the variable will contain the number of 256 byte blocks in the file. For example, if the file DATA is 30 blocks long, then

```
20 OPEN #0,"DATA",L
```

will place 30 in L after successful opening.

E) Multiple program execution. The CHAIN statement has been added to allow one program to cause another BASIC program to be loaded into the program area and executed. Execution of CHAIN statement has an identical effect to stopping the current program, loading the new program, and typing RUN. Thus all variables are re-initialized, and all open files are closed. Communication between the two programs must be done either by writing variables onto a data file or into an area of RAM using EXAM and FILL. E.g.,

```
20 CHAIN "PART2"
```

F) ARCTANGENT. The function ATN has been added for computing the arctangent of a specified value.

G) SIZE CHANGES. The above additions have expanded the size of BASIC by about 1/2K over the size of Release 2. For those who need more RAM space for programs and data, there is a method of releasing the space used by the mathematical functions. The following table shows how much ENDBAS may be reduced by to remove each of the functions listed. Note that functions must be released in the order presented (i.e., it is not possible to remove EXP and keep ATN).

```
ATN       A0 HEX
SIN-COS  130 HEX
LOG       90 HEX
EXP       90 HEX
```

We will appreciate written reports of any errors you detect in Release 3 BASIC.

RELEASE 4 SYSTEM SOFTWARE CHANGES

North Star Computers

June, 1978

These notes, in conjunction with the following documents:

North Star BASIC, Version 6, Revision 5

The North Star Disk Operating System, Version 2, Release 3

The North Star MONITOR, Version 1, Revision 5

describe the Release 4 System Software (BASIC, DOS, and Monitor) distributed beginning in June, 1978. These notes are intended to describe the improvements made since the above documents were published. A new manual, "North Star SYSTEM SOFTWARE" is in preparation. This manual contains revised, expanded, and updated documentation on BASIC, DOS, and the Monitor.

RELEASE 4 BASIC

The Release 3 version of North Star BASIC has been in use since August, 1977. The Release 3 description notes are included as an appendix to these notes for those who are not familiar with Release 3 BASIC. The proper sequence of reading is: BASIC manual, Release 3 appendix, and finally the following description.

New BASIC Commands

/PSIZE
This command will cause the size of the current BASIC program to be printed on the terminal. The value printed will be the number of file blocks (i.e., 256-byte blocks) required to store the program.

NSAVE <file name> <optional file size>
The NSAVE command will create a new type 2 file and store the current program on that file. The size of the new program may be specified (in file blocks) by the optional second argument. If the optional size is not supplied, then NSAVE will make the new file 3 blocks larger than required. Note that the optional size argument is separated from the file name by a blank character instead of a comma as other BASIC commands.
NSAVE TEMP,2
NSAVE NEW,2 25

CAT <optional # device number,> <optional drive number>
The CAT command will print the directory (catalog) on the

specified disk drive (drive 1 is the default). An optional output device number may be specified as in the LIST command.
CAT
CAT 2
CAT #2,3
CAT #2

MEMSET <memory address>
This command may be used to change the upper bound of the program and data region available to BASIC. The change will become permanent (i.e.,, if the version of BASIC is re-enterred at any entry point, the new memory size will remain). The following example sets the upper bound to the standard value used with a 16K RAM board addressed at 2000H (8192):
MEMSET 24575

AUTO <optional initial value>,<optional increment>
The AUTO command will enter automatic line number mode. In this mode, a new line number will be printed at the start of every line. The first argument specifies the initial line number (default is 10) and the second argument specifies the increment (default is 10). Auto line number mode will persist until one of the following occurs: a) a null line is entered (i.e., a carriage return is typed immediately after the line number) or, b) a command is typed (by using the editor to delete the line number from the beginning of the line).
AUTO
AUTO 1000
AUTO 100,100

APPEND <file name>
The APPEND command may be used to load the specified program at the end of the current program. Thus, concatenating two programs is possible with the APPEND command. It is an error to append a file whose first line number is less than or equal to the last line number of the current program.
APPEND TEMP,2

BASIC Personalization Bytes

ENTRY POINT #3 (2A14H,10772D)
This is a new entry point to BASIC which is similar to the 2A04 entry point, but which also will preserve the values of all variables.

AUTOS (2A0FH,10767D)
This byte allows for creating a copy of BASIC which will immediately begin execution of a BASIC program after the GO command. Proceed as follows: 1)GO BASIC and load the desired program. 2)Execute the PSIZE command. 3)Leave BASIC with BYE, and change the AUTOS byte from 1 to 0. 4)Create a file (e.g.,

GAME) with size equal to the size of BASIC plus the PSIZE value (e.g., 57). 5)Set the type to 1 and go-address to 2A00. 6)Do an SF GAME 2A00. Subsequent execution of GO GAME will be equivalent to GO BASIC; LOAD prog; RUN.

CONC (2A18H,10776D)
Changing this byte from 0 to 1 inhibits the use of control-C for program interruption. When control-C is inhibited, the CONTC routine in DOS will not be called during program execution. Note that control-C can be enabled and disabled under program control by use of the FILL statement.

PAGES (2A13H,10771D)
If PAGES is non-zero, then all LIST commands to device 0 will be done in "paging mode". In paging mode, after PAGES-1 lines have been printed, the user will be asked to type a carriage return to continue the listing.

BSPC (2A17H,10775D)
This byte specifies the character to be printed when the character-delete editing command (underline or control-Q or backspace) is typed. Users with CRT terminals may wish to change the BSPC character from underline to backspace (08H).

LINECT (2A0EH,10766D)
This byte contains the initial line length used by BASIC. The default value is 80. The maximum value allowed is 132.

LINETB (2A11H,10769D)
These two bytes contain a pointer to the "print head table" in BASIC which contains a one-byte current cursor or print head position for each of the 8 I/O devices (starting with device 0). For some applications such as plotting, some users may wish to EXAM or FILL these bytes to avoid getting the LENGTH ERROR message.

Immediately following the print head table is the file table, which contains one 10-byte entry for each of the 8 possible open files. Byte 0: status byte, bytes 1-2: buffer address, bytes 3-4: disk address of file, bytes 5-6: number of blocks in file, bytes 7-9: current file pointer.

PROM (2A10H,10768D)
This byte must be changed from 0E8H if a non-standard origin disk controller PROM is used. If this byte is not changed, then the new randomizing feature will not work.

File Processing in BASIC

Data files are no longer restricted to a maximum size of 64K bytes, and data files are no longer restricted to be of type 3. Also, up to 8 data files may be simultaneously open at any one time. The following statements have been added or extended:

CREATE <file name string>,<length><,optional type>
   This statement will attempt to create a new file with the specified name and length (given in 256-byte blocks). The type of the file may be optionally specified as the third argument, otherwise type 3 will be used.
   CREATE "TEMP",5
   CREATE "DATA,2",300,9
   CREATE B$,N,T

DESTROY <file name string>
   This statement will attempt to destroy the specified file. The file may be of any type. As with most file statements, the DESTROY statement may be executed as a direct statement.
   DESTROY "TEMP,2"

OPEN #<file number>%<file type>,<file name>,<optional variable>
   The OPEN statement has been extended so that if a % follows the file number expression in an OPEN statement, then the following expression specifies the type that the specified file must be. If the % is omitted, then the file must be of type 3.
   OPEN #N %9, "DATA,2"

The following new function has been added:

FILE(<file name string>)
   This function will return a -1 as value if the specifed disk file does not exist. Otherwise, if the file exists, then the function will return the type of the file.
   T=FILE("TEMP,2")
   T=FILE(F$)

Randomizing the BASIC Random Number Generator

If the RND function is called with a negative argument, then the random number seed will be set to a "random" starting value. [This is accomplished by measuring the time to the next disk sector pulse.] After once using T=RND(-1) to set the seed, subsequent random numbers should be obtained by using RND(0). Note that better results will be obtained if user input is requested between any disk accesses and the call on RND(-1).

Error Trapping in BASIC

A new statement has been added for allowing a BASIC program to retain control even when errors occur. To enable the error trapping mode, the statement:

ERRSET <line number>,<variable>,<variable>

should be used. The line number specifies where control should be passed when an error occurs. The two variables will be set to the line number of the offending statement, and the type code for the error, respectively. The trappable errors and their type codes are:

```
 1    ARGUMENT ERROR
 2    DIMENSION ERROR
 3    OUT OF BOUNDS ERROR
 4    TYPE ERROR
 5    FORMAT ERROR
 6    LINE NUMBER ERROR
 7    FILE ERROR
 8    HARD DISK ERROR
 9    DIVIDE ZERO ERROR
10    SYNTAX ERROR
11    READ ERROR
12    INPUT ERROR
13    ARG MISMATCH ERROR
14    NUMERIC OVERFLOW ERROR
15    STOP (on control-C)
```

Note that control-C is trappable (but may be disabled as described above if desired).

When an error occurs, error trapping is turned off, control is passed to the specified line, and the variables are set. All GOSUB, user function, and FOR NEXT histories are left intact, so that it is possible to continue after the error. Error trapping mode may be disabled by executing an ERRSET statement with no arguments.

BASIC Math Functions

The math finctions SIN, COS, EXP, and LOG have been rewritten to improve accuracy. The ATN function has not yet been changed. The math functions may be deleted as described in the Release 3 appendix.

Miscellaneous BASIC Changes

A new function INCHAR$(<device number expression>) has been added. This function will await the typing of a single character from the specified device, and return that character as a single-character string. Control characters as well as printing characters are allowed. The character will not automatically be "echoed" to the device.

```
10 T$=INCHAR$(0)\ PRINT T$,
```

The VAL function will now accept strings which have initial blanks. Note that the STR$ function does conversion based on the current default PRINT format.

If a control-N or at-sign editing character is used to edit the response to an INPUT statement that has a prompt (e.g., INPUT "AGE: ",A$), then the prompt will be retyped at the beginning of the new line.

The LIST command now has four forms:

```
LIST              list entire program
LIST 100          list specified line
LIST 100,         list specified line thru end of program
LIST 100,200      list lines indicated by range
```

Due to reassignment of reserved word values, the appearance of the word CREATE in REM statements of old programs will be changed to AUTO in Release 4. Likewise, DUMP will be changed to MEMSET and NULL will be changed to NSAVE.

The NSAVE command and CREATE statements will not work correctly if Release 4 BASIC is run on a Release 3 DOS.

Release 4 BASIC is a superset of Release 3 BASIC, and all programs that ran under Release 3 should run under Release 4.

The enhancements to BASIC have enlarged it to about 12.5K (50 block file). [The M5700 Monitor program now overlaps BASIC.] You may wish to retain your copy of Release 3 BASIC for those applications where you do not need the new features but do need to save as much memory as possible.

RELEASE 4 DOS CHANGES

Note that the following changes apply to all personalized and unpersonalized versions of the DOS.

1. Compact removed. The CO command has been removed, and now is provided as a separate GO file, called COMPACT. To compact the file space of a diskette, do GO COMPACT. COMPACT will then ask you to mount the diskette to be compacted and specify the drive number. Remember that compact may not be used on diskettes which contain any zero-length files or any overlapping files.

2. Turnkey startup. A version of the DOS may be created so that

27C0

at bootstrap startup, a specified command (e.g., GO BASIC) is immediately executed after the terminal initialization routine returns to DOS. To create such a version of DOS: 1)Load a fresh copy of DOS into RAM, 2)Change byte 28E5H from a 1 to a 0, 3)Enter the ASCII codes for the desired command beginning at byte 27C0H (followed by a carriage return), 4)Save the modified DOS on diskette at disk address 4. It should now be possible to use the modified DOS to perform the desired command immediately after bootstrap load. [For convenience,GO BASIC has already been entered, so if that is the desired command, all that must be done is to modify the flag byte.]

3. Output device specification. The LI command may take an optional output device specification, as in BASIC. An optional number sign followed by the device number may immediately follow the LI command.

```
LI #2       List drive 1 on device 2
LI #2 3     List drive 3 on device 2
LI          List drive 1 on device 0
LI 2        List drive 2 on device 0
```

4. Paging. The LI command may be personalized so that after each N lines of file directory listing, the message "TYPE RETURN TO CONTINUE" will be printed. Personalize byte 28F8 to the number of lines on your CRT screen. To disable paging mode, set the byte to 0.

Note that the personalization process described on page 14 of the DOS manual should now create a 50 block file for BASIC rather than a 45 block file.

MONITOR

The Monitor has not been changed. However, a fifth Monitor, M6700 is provided on the Release 4 diskette. The M6700 Monitor is personalized with its own I/O routines which communicate with a HORIZON standard serial port, as opposed to using the DOS interfacing conventions. The M6700 Monitor has been added because the M5700 Monitor can no longer be used to personalize the enlarged Release 4 BASIC.

As always, we will appreciate hearing as soon as possible about any problems you may encounter.

APPENDIX: RELEASE 3 CHANGES TO NORTH STAR BASIC

Release 3 BASIC fixes all reported bugs in Release 2 BASIC, and also makes several additions. Release 3 is a superset of Release 2 - all programs which run under Release 2 should run under Release 3. Note that the size of BASIC has increased - now a 45 block file is required.

A) Multiple input/output devices. Now PRINT, INPUT, LIST, and LINE may specify a "device expression" as an optional first argument preceded by a number sign, e.g.:

```
PRINT #3,A,B,C
INPUT #X,"TYPE A NUMBER: ",N
LIST #2,100,500
LINE #2,64
```

Device 0 is assumed if the device expression is missing. Also, device 0 is always used for command communication in direct mode and in the DOS. LINE becomes an executable statement instead of a command, so that device line lengths can be set up within programs as well as in direct mode.

The device value, which must be between 0 and 7 inclusive, is communicated in the ACC to the DOS CIN and COUT routines. The DOS routines can then decide which device to communicate with based on the contents of the ACC. Note that no changes are needed for the DOS I/O routines if this feature will not be used. Also note that this feature may be used to communicate string values to machine language subroutines.

B) Delete command. The DEL command has been added to allow the deletion of a block of program statements. For example,

```
DEL 100,200
```

will delete all program statements with line numbers in the range of 100 through 200 inclusive.

C) Byte access to data files. It is now possible to access a data file in BASIC at the byte level. A numeric value in the range of 0 to 255 may be written into or read from a specified byte address in a data file. These byte accesses may be performed in either sequential or random mode. To indicate byte mode, an ampersand (&) is placed before the variable to be read into or the expression to be written out. For example,

```
10 READ #0,A,&B
20 WRITE #1,&X+1,B$
30 READ #0%X,&T,&T1
40 WRITE #F%X,&ASC("a")
```

D) Determining file length. The length of a data file may be determined by adding an optional variable name at the end of the OPEN statement. After the OPEN statement, the variable will contain the number of 256-byte blocks in the file. For example, if the file DATA is 30 blocks long, then

```
20 OPEN #0,"DATA",L
```

will place 30 in L after successful opening.

E) Multiple program execution. The CHAIN statement has been added to allow one program to cause another BASIC program to be loaded into the program area and executed. Execution of the CHAIN statement has an identical effect to stopping the current program, loading the new program, and typing RUN. Thus all variables are re-initialized, and all open files are closed. Communication between the two programs must be done either by writing variables onto a data file or into an area of RAM using EXAM and FILL.

```
20 CHAIN "PART2"
```

F) Arctangent. The function ATN has been added for computing the arctangent of a specified value.

G) Size changes. The above additions have expanded the size of BASIC by about 1/2 K over the size of Release 2. For those who need more RAM space for programs and data, there is a method of releasing the space used by the mathematical functions. The following table shows how much ENDBAS may be reduced by to remove each of the functions listed. Note that the functions must be released in the order presented (i.e., it is not possible to remove EXP and keep ATN).

ATN	A0 Hex
SIN-COS	130 Hex
SIN-COS	130 Hex
LOG	90 Hex
EXP	90 Hex

We will appreciate hearing written reports of any errors you detect in Release 3 BASIC.

ERRATA AND ADDITIONAL INFORMATION

for: SYSTEM SOFTWARE MANUAL, Revision 2.1

North Star Computers, Inc.

JULY 27, 1979

This Errata sheet describes the new features of Release 5.1 system software: DOS, BASIC, Monitor, and Utilities. There are two versions of Release 5.1 software. For single density (old controller board) users, Release 5.1S software conforms, except as noted below, to the description given in the SYSTEM SOFTWARE MANUAL. The latest dual density software, Release 5.1DQ, (to be used with dual density controllers) takes full advantage of North Star's new quad capacity (double sided) disk drives, and will handle any combination of single and double sided drives.

1. The format of a double sided diskette is as follows: Side A is the same as a single sided diskette and can be used in single sided disk drives. Side B holds tracks 35 through 69 with track 35 at the inside (opposite track 34), and with track 69 at the outside (opposite track 0). The directory is still on side A, but entries in it may refer to disk addresses up through 699. On a double sided, double density diskette, a file which does not overlap the directory area may now be as large as 1392 blocks (348K bytes, or 696 sectors). Double sided diskettes may be used in single sided drives, but only the data on side A may be accessed. If a file "wraps around" to side B, only the part of it on side A can be accessed in a single sided drive. Any attempt to access the remainder of such a file will cause an error.

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

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

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

   A. The upper four bits tell which drives are double sided. Bits 7 down through 4 correspond to drives 1 through 4, respectively. Each bit must be 0 if the corresponding drive is single sided, or 1 to allow double sided operation of that drive.

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

For example, a CONFG value of 5AH (01011010) indicates that only drives 2 and 4 are double sided, fast-stepping drives. The proper CONFG value for four quad capacity drives is 0FFH. DOS on factory master diskettes is supplied with zero in the CONFG byte, because any drive can be operated at normal speed, using side A only.

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

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

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

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

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

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

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

10. Because North Star now supports parallel-interfaced printers, the I/O personalization code for the HORIZON computer has been modified. A listing of the Release 5.1 I/O code follows, and supercedes that given in chapter H of the GETTING STARTED section of the SYSTEM SOFTWARE MANUAL:

RELOCATING Z80 MACRO ASSEMBLER VERSION 2.21
.HR" -
I/O ROUTINES FOR STANDARD HORIZON IN RELEASE 5.1 DOS

```
                        .IDENT  .HRZ.
                        ;
200D                    .LOC    200DH   ;JUMP TABLE
200D    C33429          JMP     COUT
2010    C30029          JMP     CIN
2013    C37E29          JMP     TINIT
2016    C36F29          JMP     CONTC
                        ;
2900                    .LOC    2900H   ;I/O BLOCK
2900    FE02    CIN:    CPI     2       ;CHECK FOR DEVICE 2 POSSIBILITY
2902    CA2229          JZ      CIN2    ;JMP IF PARALLEL PORT SPECIFIED
2905    FE01            CPI     1       ;CHECK FOR DEVICE 1 POSSIBILITY
2907    CA1629          JZ      CIN1    ;JUMP IF SECOND PORT SPECIFIED
                        ;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 CHARACTER IN A
                        ;
2916    DB05    CIN1:   IN      5
  18    E602            ANI     2
291A    CA1629          JZ      CIN1
291D    DB04            IN      4
291F    E67F            ANI     7FH
2921    C9              RET
                        ;
                        ;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       ;RESER PI FLAG
2930    F1              POP     PSW
2931    E67F            ANI     7FH
2933    C9              RET
```

RELOCATING Z80 MACRO ASSEMBLER VERSION 2.21

I/O ROUTINES FOR STANDARD HORIZON IN RELEASE 5.1 DOS

```
2934    FE01        COUT:   CPI     1
2936    CA4929              JZ      COUT1   ;SECOND SERIAL PORT OUTPUT
2939    FE02                CPI     2
293B    CA5429              JZ      COUT2   ;PARALLEL PORT OUTPUT
                    ;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
                    ;
                    ;PRINTER 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 READY YET
295B    3E20                MVI     A,20H   ;RESET PO FLAG
295D    D306                OUT     6
295F    78                  MOV     A,B     ;CHARACTER TO ACC
2960    F680        TIN1:   ORI     80H     ;SET STROBE FALSE
2962    D300                OUT     0       ;SEND CHARACTER
2964    EE80                XRI     80H     ;TOGGLE STROBE
2966    D300                OUT     0
2968    EE80                XRI     80H     ;TOGGLE STROBE
296A    D300                OUT     0
296C    E67F                ANI     177Q    ;MASK DOWN TO ASCII CHAR
296E    C9                  RET
                    ;
296F    DB03        CONTC:  IN      3       ;INPUT SERIAL PORT STATUS
2971    E602                ANI     2       ;MASK INPUT STATUS BIT
2973    EE02                XRI     2       ;SET Z-FLAG ONLY IF CHARACTER
2975    C0                  RNZ             ;RETURN IF NO CHARACTER TYPED
2976    DB02                IN      2       ;INPUT THE CHARACTER
2978    E67F                ANI     7FH     ;MASK OFF PARITY BIT
297A    FE03                CPI     3       ;SEE IF CHARACTER IS CONTROL-C
297C    37                  STC             ;TELL SOFTWARE A CHAR WAS TYPED
297D    C9                  RET             ;RET WITH Z-FLAG PROPERLY SET
```

RELOCATING Z80 MACRO ASSEMBLER VERSION 2.21

I ROUTINES FOR STANDARD HORIZON IN RELEASE 5.1 DOS

```
                        ;TINIT FIRST REWRITES ALL RAM TO SET PARITY CORRECT
297E    3E40            TINIT:  MVI     A,40H   ;DISABLE PARITY LOGIC
2980    D3C0                    OUT     0C0H    ;BEFORE READING UNWRITTEN RAM
2982    210000                  LXI     H,0     ;PREPARE TO CYCLE THROUGH RAM
2985    16E8                    MVI     D,0E8H  ;SET UP TO SKIP DISK REGION
2987    7C              TINKL:  MOV     A,H     ;MOVE CURRENT BLOCK NUMBER TO A
2988    BA                      CMP     D       ;CHECK IF DISK BLOCK
2989    C29229                  JNZ     TINCP   ;CONTINUE IF NOT DISK BLOCK
298C    C604                    ADI     4       ;ADD 1K TO RAM ADDRESS
298E    67                      MOV     H,A     ;PUT UPDATED ADDRESS BACK TO HL
298F    CAA529                  JZ      TINU    ;MIGHT BE DONE IF NON-STANDARD
2992    7E              TINCP:  MOV     A,M     ;READ BYTE FROM RAM
2993    77                      MOV     M,A     ;RESTORE IT WITH CORRECT PARITY
2994    2C                      INR     L       ;INCREMENT LOW ORDER ADDR BYTE
2995    C29229                  JNZ     TINCP   ;LOOP IF NOT END OF 256 BLOCK
2998    24                      INR     H       ;INCREMENT BLOCK NUMBER
2999    CAA529                  JZ      TINU    ;DONE IF WE ARE BACK TO ZERO
299C    7C                      MOV     A,H     ;BLOCK NUMBER TO A
299D    E603                    ANI     3       ;MASK LOW ORDER 2 BITS
299F    C29229                  JNZ     TINCP   ;CONT IF NOT END OF 1K BLOCK
29A2    C38729                  JMP     TINKL   ;BRANCH TO MAIN LOOP
                        ;
                        ;NOW THAT ALL PARITY BITS CORRECT, ENABLE PARITY LOGIC
                        ;
29A5    3E41            TINU:   MVI     A,41H   ;PARITY ENABLE CODE
29A7    D3C0                    OUT     0C0H    ;MEMORY BOARD OUTPUT PORT
                        ;
                        ;NOW INITIALIZE MOTHERBOARD AND SERIAL PORTS
29A9    AF                      XRA     A       ;ZERO ACC
29AA    D306                    OUT     6       ;INITIALIZE MOTHERBOARD
29AC    D306                    OUT     6       ;EXTRA
29AE    D306                    OUT     6       ;EXTRA
29B0    D306                    OUT     6       ;EXTRA
29B2    3ECE                    MVI     A,0CEH  ;2 STPS, 16xCLK, 8 BITS, NO PAR
29B4    D303                    OUT     3       ;SEND TO FIRST SERIAL PORT
29B6    3ECE                    MVI     A,0CEH  ;SAME CODE AS FIRST PORT
29B8    D305                    OUT     5       ;SECOND PORT
29BA    3E37                    MVI     A,37H   ;CMD: RTS, ER, RXF, DTR, TXEN
29BC    D303                    OUT     3       ;FIRST PORT
29BE    3E37                    MVI     A,37H   ;SAME CODE AS FIRST PORT
29C0    D305                    OUT     5       ;SECOND PORT
29C2    DB02                    IN      2       ;CLEAR SERIAL INPUT BUFFERS
29C4    DB04                    IN      4
29C6    3E30                    MVI     A,30H
29C8    D306                    OUT     6       ;RESET PI FLAG
                        ;
                        ;NOW INITIALIZE PRINTER ON PARALLEL PORT
29CA    3E60                    MVI     A,60H   ;CODE TO SET PO FLAG
29CC    D306                    OUT     6
29CE    3E0D                    MVI     A,13    ;CARRIAGE RETURN
29D0    C36029                  JMP     TIN1    ;GO OUTPUT CR TO PARALLEL PORT
                        .END
```