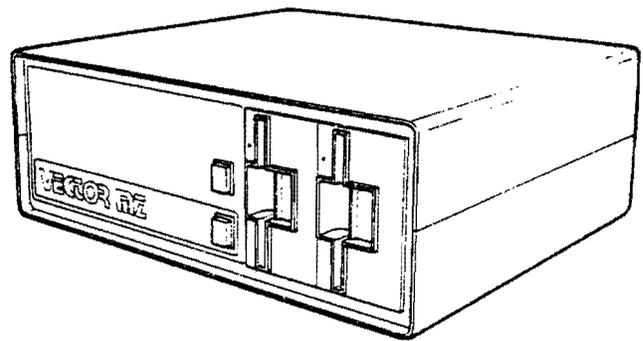
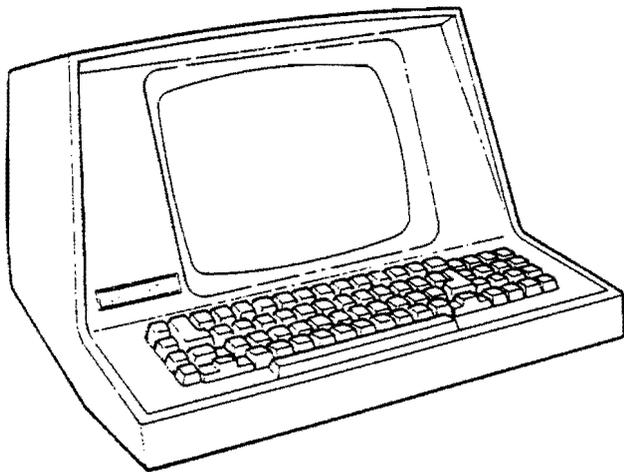


MZOS

Users Guide



M Z O S

MZ Operating System

by Vector Graphic Inc.

(C) 1978 Vector Graphic Inc.

Table of Contents

Introduction	1
File Structure	2
Memory Layout	4
Console Commands	5
Usage Notes	8
Errors and Mistrakes	9
Famous Last Words	11
Appendix A: Entry Points	12
Appendix B: Console I/O	14
Appendix C: Listing Format	18

INTRODUCTION

The Vector MZ operating system, MZOS, was designed specifically for the MZ system. It is a file-oriented disk operating system, allowing you to maintain and use files on the disk. Also provided are subroutines which may be used by the assembly language programmer to interface other software to the disk system, or for programs like BASIC to load and save files on the disk.

MZOS is a copyrighted software product of Vector Graphic Inc. and is meant only for use on an MZ or similar system from Vector Graphic. We assume no responsibility for unauthorized use.

This manual reflects version 1.5 of MZOS.

FILE STRUCTURE

First, it is necessary to understand the layout of the disk and the files on it. Thorough understanding of this section will greatly enhance ease of operation of this system.

The disk contains 77 concentric tracks, similar to the grooves in a record. Each track is divided into 16 sectors, and each sector contains one page (256 bytes) of data. Thus the capacity of the entire disk is around 315K bytes. The tracks are numbered 0-76, and the sectors 0-15. For our purposes, however, the disk should be thought of as 1232 sectors, numbered 0-1231, forgetting about tracks altogether.

The first four sectors on the disk are reserved for the directory. The purpose of the directory is to keep track of the files on the disk. There are 64 entries allowed, therefore you may have as many as 64 files on the disk. Each entry in the directory uses 16 bytes. The format of each entry is as follows:

<u>FILENAME</u>	<u>ADDRESS</u>	<u>LENGTH</u>	<u>TYPE</u>	<u>START</u>	<u>VERSION</u>
0-7	8-9	10-11	12	13-14	15

The meanings of each field are as follows:

FILENAME

An eight-byte field, this contains the name of the file. It may contain any printing ascii character, expect spaces, commas, or lower-case letters.

ADDRESS

This two-byte field contains the address on the disk where the file begins. This address is actually just the sector number of that particular sector. Sector 278 would have a disk address, then, of 278.

LENGTH

The length of the file is the number of sectors it occupies.

TYPE

The type of a file is a special reference to what the file might be used for. A file may be of type 0 through 99. Certain file types have already been defined. They are:

0	Default type unless changed.
1	Executable machine language program.
2	BASIC program.
3	BASIC data file.
4	Ascii text file.
5-7	Reserved.
8	XEK Assembler source file
9	Reserved.

File types 10-99 are not currently used or reserved, so they can be used for special purposes.

START

The start bytes are used to define the loading/starting address for type 1 (machine language) files. In the case of BASIC files, they're used to define the amount of valid data within the file.

VERSION

This byte is used to define the version or revision of the file. This is used by the Memortite II system, but it not implemented other than that.

The remaining 1228 sectors on the disk are available for files. There is a restriction that a file may only be 256 sectors long. This is not unreasonable, since it would be rather difficult to load a file longer than 64K into memory anyway. Actually, you may have a file longer than 256 sectors, but you will not be able to load, save, or verify it with MZOS.

A fact to remember is that MZOS does not necessarily access a file when it accesses the directory. Create, delete, and similar operations only modify the directory, not the actual file. Thus if you accidentally delete a file, you can recover it just by creating it with the same addresses as it had before. If you need to create a bigger file, you can just delete the old one and then recreate it with a greater length and the same starting address.

Another thing to remember is that MZOS maintains no copies of the directory in memory, so you may exchange disks at will. (This is as opposed to, say, CPM.)

MZOS MEMORY LAYOUT

It is important to know the places in memory used by MZOS. There are actually three separate areas to know about. The first, symbolically called MZOS, is the actual operating system and buffer areas. This resides in RAM from 2000-29FF. The top of this area is used for system i/o and is explained later. The second area is called MZIO and is a 1K prom at address C400-C7FF. This is the second 2708 on the prom/ram board. It contains all of the disk i/o routines. The third area, MZTMP, is an approximately 32-byte block starting at DF40, in the ram area of the prom/ram board. This is used for the track table and similar important information.

In addition to the regular area, a 2K area immediately following MZOS (actually 2A00-31FF) is used for some mass transfer commands to achieve a higher speed. Execution of the IN, DT, CF, CD, or CO commands will utilize this area, thus overwriting whatever is there.

Last but not least, there are several entry points to the system which you should know about. These, as well as the disk subroutines, are covered in appendix A.

MZOS CONSOLE COMMANDS

Following is an explanation of MZOS commands and their usage. The following conventions are used in this explanation: When something is enclosed in parentheses, e.g. (address), that argument is optional. Something in angle-brackets, e.g. <address>, means that you should substitute a valid argument; in this example, you should type an address. Also, all arguments are in decimal, with the exception of memory addresses, which are in hex.

LI (<unit>)

This will list the directory on the specified unit, with the current disk as the default. Each file will be listed, followed by the address, length, protect status, type, and start address if the file is type 1. A sample listing is shown in appendix C.

FL (<unit>)

This is similar to the LI command, only the directory is listed in a 'fast' format. Only the file is listed, six-wide across the page. This is useful for seeing just if you have a particular file.

CR <file> <length> (<address>)

The create command allows you to create a file entry in the directory. The file will be created with the specified length. If an address is specified, it will be used, otherwise the first free disk address will be defaulted to.

DE <file>

This will delete a file entry. Remember, the file itself will not be harmed, just the entry in the directory. A protected file cannot be deleted.

RN <old-file> <new-file>

This allows you to change the name of (rename) a file. All the other information (type etc.) will remain the same. Do not rename a file to a name that already exists!

TY <file> <type> (<start addr>)

This sets the type of a file. If type 1 is specified, the start address **MUST** be specified. No accessing of the file is actually done, just the directory entry.

PR <file>

This protects the file specified. When a file is protected, it cannot be deleted. The file may still be written into, however! This is not a perfect protection, but it does keep you from deleting an important file in a moment of frustration. **IMPORTANT NOTE!** Assembler source (type 8) and BASIC (types 2 and 3) files **MAY NOT** be protected. If you do, any attempt by XEK or BASIC to look them up will fail.

UP <file>

This will just unprotect a file. No error is caused by unprotecting a file that wasn't protected anyway.

LF <file> <mem addr>

SF <file> <mem addr>

VF <file> <mem addr>

These commands load, save, or verify a file. The load and save commands do so with respect to the memory address specified. Thus 'SF TEST 3000' would save memory into a file called TEST, starting at memory address 3000. Data will be written into the file sequentially until it is full. The LF command works in a similar manner, only data is read from the disk into memory. The VF command does no ram access; it verifies the file by doing an internal check on it. The memory address is necessary, though, so just use 0 as a dummy address,

CF <old-file> <new-file>

This will copy the data from one file to another. In addition, the type and protect information will be copied too. Make sure that the destination file is at least as large as the source file, or you will receive an error.

GO <file>

This will load a type 1 file into memory at its start address and jump to it. Note that this is equivalent to using the LF command followed by a JP command to the start address of the file. Obviously, this only is for type 1 files.

RD <addr> <mem addr> <sectors>

WR <addr> <mem addr> <sectors>

VR <addr> <mem addr> <sectors>

These commands will read, write, or verify data directly on the disk, with no regard for files. These commands are used infrequently, as normally all work is done with files. Using the RD command as an example, take the command RD 4 3000 10. This will read 10 sectors, starting with address 4, and load them into memory beginning at address 3000 (hex). The WR and VR commands work similarly. The VR command, like the VF command, does internal checking only. You may use 0 as a dummy ram address for this command.

IN (<unit>)

This will initialize a fresh diskette. It completely fills the diskette with spaces (ascii 20 hex). This **MUST** be done to **ALL** diskettes before use, as they contain garbage before initialization. This command also writes the sector id's on each of the sectors (These id's are used for error checking). You shouldn't initialize a diskette that contains valid data, of course.

DT (<unit>)

This will test a diskette by writing and then verifying a constantly changing pattern over the entirety of the diskette. This destroys any data that was on the diskette, so use with caution, and **NEVER** on a diskette after it has valid data written on it.

CD <source> <dest>

This will copy the entirety of the source diskette onto the destination diskette. No file-oriented access is used, just direct read and write. The resulting diskette is an exact duplicate of the source diskette.

CO (<unit>)

This compacts the data on the disk by moving all files toward the beginning of the disk. This can become necessary when you have several files, and you delete one or more out of the middle. See also a note in the usage section of this manual referring to this command.

JP <mem adr>

This just transfers control to the specified address in memory. It is used, for example, to start a program that is already in memory.

DD (<unit>)

This sets the default unit to the one specified. If no unit is specified, then unit 1 is used. All disk accesses, when no unit is specified, use the default unit.

; <text>

Typing a semicolon (;) as the first character of a line will cause the line to be ignored. Only thirty characters may be typed, though, or you will receive an error.

<file>

Typing a file name alone is exactly like typing GO followed by the file name. Thus BASIC and GO BASIC are identical commands.

USAGE NOTES ON MZOS

The following is a collection of things to know while using MZOS.

When typing a command, it may be aborted at any time by typing control-C. To erase the last character typed, hit Backspace, Underscore, or DEL.

During execution of a command like DT or IN, you may abort execution by typing control-C.

You should not have any type 1 files with a filename less than three characters long. If you do, you will have to use the GO command to run it.

The DT command will run until it finds an error or you stop it.

The IN command takes around a minute and a half, or about two and a half if you have the read-after-write check enabled.

NEVER NEVER NEVER NEVER interrupt the execution of the CO command, unless you don't mind your diskette being totally destroyed. Compaction normally takes well under five minutes, but depending on the degree of disaster your diskette is in, it could take ten or twenty minutes to compact it. DON'T PANIC! A relatively messy disk could require over a million read/writes, and interrupting it in midstream is guaranteed to leave the disk in a totally unknown state. So unless the disk starts smoking or making really terrible sounds, just leave it alone.

In a multiple-unit system, the particular drive which you wish to reference is indicated by adding a ,1 ,2 ,3 or ,4 to either a filename or disk address. Thus referencing the file TEST on unit 2 would look like TEST,2; a reference to sector 24 on unit 3 would be 24,3. This is true of ANY file reference, in any command; or of the disk address in the RD, WR and VR commands.

MZOS will digest lower case letters as well as upper case, since it just translates everything to upper case anyway.

To repeat something said elsewhere, all disk accesses, where an explicit unit isn't specified, reference the disk specified in the last DD command. If no such command has been issued, then unit 1 is used.

If you have a printer interfaced with your system, you should know that error messages will only print on the console, not the printer, even if the printer is enabled. This prevents getting error messages in the middle of a good printout.

Needless to say, this operating system is completely compatible with the North Star DOS. It is specifically designed so that any software that runs on the North Star system will run on MZOS.

ERRORS AND MISTRAKES

A reasonably comprehensive set of error messages are provided with MZOS. Following is a list of these errors and why you might get them.

Huh?

This means that you typed something that MZOS can't possibly understand. Usually this is from misspelling a command, or typing an invalid filename as an implied GO.

% Syntax error

This indicates that you typed a valid command, but invalid arguments. Examples: TY FILE 1 without a start address, or LF FILE QWERT where QWERT is obviously not a hex address.

% File error

This is caused by an invalid file reference. It can be caused by creating a file that already exists, referencing one that doesn't, trying to GO a non-type-1 file, and so forth.

% Disk overflow

You tried to create a file that would extend beyond the boundaries of the disk (that is, past sector 1231).

% Write protected

You either tried to write on a protected disk, or tried to delete a file that is protected.

% Disk offline

This can be caused by trying to access the disk when either there is no diskette in the drive, the drive is not up to speed, or the controller just isn't installed in the computer.

% Illegal argument

An attempt was made to read, write, or verify beyond the boundaries of the disk. This is similar to the overflow error, only this refers to access attempts, rather than file creation.

% CRC error at sector xxxx,n

Bad data was encountered on the disk in unit n, at sector xxxx. This is usually caused by a faulty diskette, or access to an uninitialized diskette.

% Sector id error at sector xxxx,n

This means that the sector read was not the one wanted. The most common cause of this is a glitch in the stepper motor, or possibly a bad diskette.

FAMOUS LAST WORDS

MZOS has been thoroughly tested, and should hopefully serve you well on your system. In the event you have any trouble, though, or need help in figuring out how it works, feel free to contact us for assistance. We would also appreciate any comments, suggestions, or improvements to this manual, or additions to or ideas for MZOS, that you might think up.

APPENDIX A: MZOS ENTRY POINTS

Assuming you have some familiarity with assembly language, this should help you understand interfacing to MZOS. Although MZOS is actually a Z-80 program, these explanations will use 8080 code whenever possible as it seems to be more popular.

MZOS 2028

The MZOS entry point is where you jump to start MZOS. The monitor 'J' command will jump here very nicely. At the end of some other program, putting a JMP 2028H will return control to MZOS.

MBOOT C400

This is where to jump to in order to boot up MZOS. The monitor 'E' command can be used to initially do so.

DCOM C402 or 2022

This is the nitty-gritty disk i/o routine. All disk access can be done here. Basically, what you do is set up the registers for what you want to do, then call here. (The difference is that 2022 will do the read-after-write check, and it will return to MZOS after printing any error, whereas C402 does not, and returns as explained below. The registers should be set up as follows:

A number of sectors to access
 B command - see below
 C unit - 1, 2, 3, or 4
 DE ram address to read into or write from
 HL disk address (sector) to begin access at

Commands are: 0 write from memory to disk
 1 read from disk to memory
 2 verify CRC internally on disk
 3 seek (go to) sector specified only
 4 recalibrate drive (go to sector 0)

The last two, seek and recalibrate, are mainly for test purposes. The recalibrate is used to insure proper positioning of the head, in case (for example) the track table is accidentally destroyed, or the stepper motor doesn't.

On return, if the carry flag is cleared, then all is well; otherwise, the flag will be set, and the error code will be in A' (A prime). The error codes are as follows:

0 - no error	1 - CRC error	2 - sector id error
3 - write protected	4 - disk offline	5 - illegal argument

Calling address C404 will print the error associated with the error code.

DLOOK 201C

This is used to lookup a file in the directory. Looking up a blank entry will locate a free space, for creating a new file. When this is called, A should contain the unit number, and HL should point to (contain the address of) a string of characters representing the file name, followed by either a return (0D hex) or a blank (20 hex).

When this routine returns, if the CY flag is set, then the file you looked up does not exist. In this case, HL contains the first free address on the disk. If CY isn't set, then the file was successfully found. In this case, HL points to the eighth byte of a copy of the entry. This copy is actually within the directory buffer of MZOS. See the file structure section of this manual for the structure of the entry.

DWRIT 201F

This will write a directory entry back to the disk. It is important that NO DISK ACCESS OCCUR between DLOOK and DWRIT. Now, the procedure for reading a file would be to use DLOOK to lookup the file; assuming it is really there, incrementing HL will make it point to the starting disk address of the file; then use DCOM to actually read the file. To write a new file to disk, you would first lookup the file, to make sure that it doesn't already exist. This will fail, assuming the name isn't there yet. Next, lookup a blank name. In the event this fails (it shouldn't), the disk you're using doesn't have a directory on it, or is full. Anyway, now HL points to the eighth byte of a blank entry. Now you should copy in the file name, disk address (from your first lookup), length, type, and start address (if needed). This done, call DWRIT to update the directory.

DLIST 2025

This will print the directory of the unit specified in A. The list is exactly the same as the LI command.

RWCHK 202B

This is not an entry point, but rather a flag. If this byte is 1, then a verify will be done after a write. This will slow down write operations considerably, but may be desired if you don't trust your diskette.

PRMPT 206B

This location contains the character used as a prompt. Currently it is set to a number sign (#, 23 hex), but you may change it if you like.

APPENDIX B: CONSOLE I/O

Here we will discuss the console i/o provided with your system, and how you may change it if necessary.

First, there are four entry points in MZOS which reference console i/o. They are as follows.

INCH 2010

This will input a single character from the console.

OUTCH 200D

This will output a single character.

OUTPR 200A

This will output a character to the printer.

CHKCH 2016

This does three things. First, it checks whether a character has been typed. If not, it returns immediately with the Z flag cleared. If a character has been typed, it sees if it was a control-C. If this is the case, then it returns, with Z set. Finally, if a character was typed, and it was a space, then another character is waited for. This allows you to momentarily suspend output by hitting the space bar. After another character is typed, this routine returns, with the Z flag set if that character was a control-C.

TINIT 2013

This is used to initialize the i/o system, if needed. It is currently set up to initialize the Bitstreamer board, but may be able to be replaced with RET instruction.

So. Those are the entry points. The actual routines, though, are located in a 64-byte block from 2900-29FF. A listing is provided of the routines as provided, in case your system is different.

Checking with the listing, notice that there are addresses assigned to each routine. They are spaced far enough apart that you should be able to fit in any of your own routines without moving any other routine. This means that you shouldn't have to patch the jump table (entry points) at all.

Notice that the OUTCH routine does more than just output a character to the console. Since the character to be printed is passed in the B register, the A register is used as a channel number. In the configuration provided, channel 1 is used to send output to the printer, and any other channel sends output through the monitor.

Also, notice that the input routine converts the characters 5F (underscore) and 7F (DEL) to 08 (backspace).

At this point it might be wise to point out that normal console i/o takes place through the prom monitor. PTCN (C098) is used for output, and CNTLC (C0DC) is used for input.

As for rules of use, they are as follows. (Except as specified, no registers may be changed.)

INCH returns the character typed in A.
OUTCH prints character in B; returns with it in A also.
OUTPR exactly as OUTCH, but character is sent to printer.
CHKCH may use A only; character returned is indeterminate.
TINIT currently uses A only, but may use all registers.

In the event you change the i/o routines, the procedure for updating the disk is as follows. First, you should assemble your i/o routines to run at 2900 (hex). You can overlay them directly on MZOS, providing you do no i/o while the routines are being loaded. Once overlaid, they are immediately effective, so you can check them to make certain they work. Once that's done, you can resave MZOS on disk simply by typing SF MZOS 2000.

```

2900          0001 * MZOS STANDARD I/O SYSTEM
2900          0002 * NEALE BRASSELL [15-DEC-78]
2900          0003 *
2900          0004 IOLOC EQU 2900H
2900          0005 MZOS EQU 2028H
2900          0006 *
2900          0007 *
2900          0008 * THIS DOES AN INIT, THEN JUMPS TO MZOS
2900          0009 *
2900          0010          ORG IOLOC
2900          0011 *
2900 CD 10 29 0012 REENT CALL IN18
2903 C3 28 20 0013          JMP MZOS
2906          0014 *
2906          0015 *
2906          0016 * THIS IS THE INIT ROUTINE
2906          0017 *
2906          0018          ORG IOLOC+16
2910          0019 *
2910 AF 0020 IN18 XRA A
2911 D3 03 0021          OUT 3
2913 D3 03 0022          OUT 3
2915 D3 03 0023          OUT 3
2917 3E 40 0024          MVI A,40H
2919 D3 03 0025          OUT 3
291B 3E CE 0026          MVI A,0CEH
291D D3 03 0027          OUT 3
291F 3E 27 0028          MVI A,27H
2921 D3 03 0029          OUT 3
2923 C9 0030          RET
2924          0031 *
2924          0032 *
2924          0033 * THIS CHECKS FOR <@C>, AND SUSPENDS OUTPUT ON <SPACE>
2924          0034 *
2924          0035          ORG IOLOC+64
2940          0036 *
2940 CD DC C0 0037 CHK8 CALL 0C0DCH
2943 FE 03 0038          CPI 3
2945 C8 0039          RZ
2946 FE 20 0040          CPI 32
2948 C0 0041          RNZ
2949 CD 60 29 0042          CALL IN8
294C FE 03 0043          CPI 3
294E C9 0044          RET
294F          0045 *
294F          0046 *
294F          0047 * THIS IS THE CONSOLE INPUT ROUTINE
294F          0048 *
294F          0049          ORG IOLOC+96
2960          0050 *
2960 CD DC C0 0051 IN8 CALL 0C0DCH
2963 CA 60 29 0052          JZ IN8
2966 FE 5F 0053          CPI 5FH
2968 CC 71 29 0054          CZ MAKE8

```

```

296B FE 7F          0055      CPI    7FH
296D CC 71 29      0056      CZ     MAKES8
2970 C9            0057      RET
2971 3E 08        0058 MAKES8 MVI   A,8
2973 C9            0059      RET
2974              0060 *
2974              0061 *
2974              0062 * CONSOLE OUTPUT - IF A=1 THEN THE PRINTER IS USED
2974              0063 *
2974              0064      ORG    IOLOC+144
2990              0065 *
2990 FE 01        0066 OUT8  CPI    1
2992 CA C0 29     0067      JZ     OUTPR
2995 78           0068      MOV   A,B
2996 C3 98 C0    0069      JMP   0C098H
2999              0070 *
2999              0071 *
2999              0072 * PRINTER OUTPUT - JUST USES CONSOLE
2999              0073 *
2999              0074      ORG    IOLOC+192
29C0              0075 *
29C0 78          0076 OUTPR MOV   A,B
29C1 C3 98 C0    0077      JMP   0C098H
29C4              0078 *

```

SYMBOL TABLE

```

CHK8  2940      IN8   2960      INI8  2910      IOLOC 2900      MAKES8 2971
MZOS  2028      OUT8  2990      OUTPR 29C0      REENT  2900

```

APPENDIX C: SAMPLE DIRECTORY LISTING

Here is a sample listing, as produced by the LI command.

MZOS	4	10	P	00	BASIC	14	45	01	2A00
DEX	59	23	P	01 2A00	TESTFL	82	4	02	
IOSYS	86	90		08	DATAFILE	176	250	03	
INXFILE	426	50		03					

Here is an analysis of the above listing.

MZOS is a file, starting at address (sector) 4 and taking 10 sectors. It is protected, so it cannot be deleted.

BASIC is a 45-sector machine code file, starting at address 14. It's ram starting address is 2A00. It could be executed by typing 'GO BASIC', or just 'BASIC'.

DEX is a machine code file, similar to BASIC. It is protected, though. Its starting ram address is also 2A00.

TESTFL is a BASIC program (type 2), which can be loaded and executed with BASIC.

IOSYS is a XEK source file (type 08).

DATAFILE is a rather large BASIC data file (type 3). It is accessed with READ and WRITE statements in BASIC.

INXFILE is a BASIC data file, like DATAFILE.

MZOS Utilities

by Vector Graphic Inc.

There are two utility and three printer driver programs provided on your system disk as received from Vector Graphic. They are NS2MZ, TITLE, DIAB, CENT, and TTY. Here we will provide you with source listings and instructions for their use.

First for the printer drivers. DIAB is the routine for a Diablo printer. It assumes that you are using ports 2 and 3 for the printer, and that the printer is the version that runs at 1200 baud, with handshaking logic.

CENT is for a Centronics printer, such as 781, 702, etc. with parallel handshaking logic. Port 1 is used for this printer.

TTY is for most serial printers, such as a TTYS, Decwriters, TI810s, and so forth. It also assumes ports 2 and 3 are being used.

For all three routines, they may be invoked by simply typing the name of the one you want, either DIAB, CENT, or TTY. If you want to save MZOS with one of these routines incorporated into it, you would just type SF MZOS 2000 after loading the driver. Example:

```
#DIAB
#SF MZOS 2000
```

The system on the diskette now has the driver incorporated in it.

Notice (in the source listings) that the various routines are spaced over the entire 2900-29FF block allocated to them. This is so that you can change the drivers without changing the jump table in MZOS. The current assignments are as follows:

REENTRY	2900	jump back to MZOS
TINIT	2910	initialize
CCCHK	2940	control-C check
INCH	2960	input character
OUTCH	2990	output character
OUTPR	29C0	output character to printer

The reentry spot is so that this can be executed as a program, allowing you to type 'DIAB' instead of 'LF DIAB 2900'.

There are a couple of features included in the input and output routines to complement the printer. First, when you type a control-P, the input routines toggles the printer flag, then discards the character. This way, even if your program doesn't allow control characters, typing control-P will still work. What toggling the printer flag actually does is to allow output to go to both the console and the printer. Typing control-P a second time will turn the printer off, etc. Second, typing control-L will send a formfeed to the printer, and discard the character. This should be done before you print something the first time, as it also sets the line counter. The output is paged; every 56 lines, it skips to the top of the next page. Checking the listings provided should help you understand how the system works.

Also, the input routine converts the Underscore character (5F) and the DEL character (7F) to Backspace (08). This is so that any one of them will work properly to erase the last character typed.

The utilities provides are NS2MZ and TITLE.

The first utility is the NS2MZ program, which, as its name implies, transfers files from North Star disk to MZOS. It is a simple program, as you can see from the listing. To use it, first initialize a disk with MZOS. Then load the NS2MZ program into memory, anywhere EXCEPT from 2000 to 3400. The program is relocatable, so it doesn't matter; we recommend that you load it at 4000. Next, boot your North Star DOS. Insert the disk you want to copy into drive 1 (North Star), and the disk you just initialized into drive 1 (Micropolis). Now JP to whatever address you loaded the program at. It will copy the entire disk, exactly as it is, onto the Micropolis disk. Now insert a system disk into the Miropolis drive and boot MZOS. Type LF MZOS 3000 (load MZOS into memory), then insert the new diskette; type SF DOS 3000 (which saves it onto the disk) followed by RN DOS MZOS (which renames the DOS to MZOS). You now have your North Star diskette on Micropolis disk. This procedure may be repeated for each disk you wish to copy. Note, though, that the disk is copied onto another disk exactly, so you may only copy disks one-to-one. Since the Micropolis disks hold more, you may want to merge several disks together manually after you've copied them.

The other utility, TITLE, titles the disk. The title is 10 characters long maximum, and is printed when you boot the disk. That is, only disks with MZOS on them can be titled. To run the program, just type TITLE. The program will print > indicating it is reading in the first sector of MZOS, then print : indicating that it is waiting for the title. At this point, type the desired title. Type carefully, since it is absolutely unforgiving of errors. After you've typed the title, hit CR. The program will print < indicating it is writing the data back out to disk, then it will return to MZOS.

Following are the source listings for all five of these programs.

```

0000          0001 *
0000          0002 * NORTH STAR TO MZOS DISK COPY PROGRAM
0000          0003 * NEALE BRASSELL [26-JUN-78]
0000          0004 *
0000          0005 * THIS PROGRAM IS 100% RELOCATABLE.
0000          0006 *
0000          0007 MZ    EQU    0C402H        ;MZOS DCOM ROUTINE
0000          0008 NS    EQU    02022H        ;NORTH STAR DCOM ROUTINE
0000          0009 *
0000          0010 DJNZ  EQU    10H
0000          0011 *
0000 3E 0A    0012 NS2MZ MVI    A,10          ;10 SECTORS AT A TIME
0002 06 23    0013          MVI    B,35          ;35 SUCH TRANSFERS
0004 0E 01    0014          MVI    C,1           ;WE'LL USE DISK 1
0006 11 00 2A 0015          LXI    D,2A00H        ;TYPICAL BUFFER SPOT
0009 21 00 00 0016          LXI    H,0           ;START FROM THE BEGINNING
000C C5       0017 N2M   PUSH   B             ;SAVE COUNTER
000D 06 01    0018          MVI    B,1           ;READ FROM NS
000F F5       0019          PUSH   PSW          ;
0010 D5       0020          PUSH   D             ;
0011 E5       0021          PUSH   H             ;
0012 CD 22 20 0022          CALL   NS           ;<< DO IT >>
0015 E1       0023          POP    H             ;
0016 D1       0024          POP    D             ;
0017 F1       0025          POP    PSW          ;
0018 06 00    0026          MVI    B,0           ;WRITE TO MZ
001A CD 02 C4 0027          CALL   MZ           ;<< DO IT >>
001D D5       0028          PUSH   D             ;SAVE DE
001E 11 0A 00 0029          LXI    D,10          ;TEN SECTORS DONE
0021 19       0030          DAD    D             ;REFLECT THIS
0022 D1       0031          POP    D             ;RESTORE DE
0023 C1       0032          POP    B             ;GET COUNTER
0024 10       0033          DB     DJNZ          ;USE NEAT Z-80 CODE
0025 E6       0034          DB     N2M-$-1        ;TO EFFECT A LOOP
0026 C9       0035          RET                    ;HOPEFULLY RETURN TO WHOEVER
0027          0036 *

```

SYMBOL TABLE

```

DJNZ  0010    MZ    C402    N2M    000C    NS    2022    NS2MZ 0000

```

0000		0001	* LITTLE PROGRAM TO TITLE A DISK	
0000		0002	* NEALE BRASSELL [12-JUL-78]	
0000		0003	*	
0000	AF	0004	TITLE XRA A	
0001	06 3E	0005	MVI B, '>'	
0003	CD 0D 20	0006	CALL 0200DH	;OUTPUT NOTE
0006	3E 0A	0007	MVI A, 10	
0008	01 01 01	0008	LXI B, 101H	
000B	11 00 2B	0009	LXI D, 2B00H	
000E	21 04 00	0010	LXI H, 4	
0011	CD 22 20	0011	CALL 02022H	;READ IN MZOS
0014	06 3A	0012	MVI B, ':'	
0016	CD 0D 20	0013	CALL 0200DH	;PROMPT
0019	21 00 2B	0014	LXI H, 2B00H	
001C	06 09	0015	MVI B, 9	
001E	CD 10 20	0016	LOOP CALL 2010H	;GET CHARACTER
0021	47	0017	MOV B, A	
0022	CD 0D 20	0018	CALL 0200DH	
0025	FE 0D	0019	CPI 13	
0027	28	0020	DB 28H	;IF <CR>, END
0028	04	0021	DB CR-\$-1	
0029	77	0022	MOV M, A	
002A	23	0023	INX H	
002B	10	0024	DB 16	;CONTINUE
002C	F1	0025	DB LOOP-\$-1	
002D	2B	0026	CR DCX H	
002E	3E 80	0027	MVI A, 128	
0030	B6	0028	ORA M	
0031	77	0029	MOV M, A	
0032	06 3C	0030	MVI B, '<'	
0034	AF	0031	XRA A	
0035	CD 0D 20	0032	CALL 0200DH	
0038	3E 0A	0033	MVI A, 10	
003A	01 01 00	0034	LXI B, 1	
003D	11 00 2B	0035	LXI D, 2B00H	
0040	21 04 00	0036	LXI H, 4	
0043	CD 22 20	0037	CALL 2022H	;WRITE MZOS BACK
0046	C3 28 20	0038	JMP 2028H	

SYMBOL TABLE

CR 002D LOOP 001E TITLE 0000

```

2900          0001 * MZOS DIABLO I/O SYSTEM
2900          0002 * NEALE BRASSELL / R.S. HARP [15-DEC-78]
2900          0003 *
2900          0004 IOLOC EQU 2900H
2900          0005 MZOS EQU 2028H
2900          0006 *
2900          0007 *
2900          0008 * THIS DOES AN INIT, THEN JUMPS TO MZOS
2900          0009 *
2900          0010          ORG IOLOC
2900          0011 *
2900 CD 10 29 0012 REENT CALL INIS
2903 C3 28 20 0013          JMP MZOS
2906          0014 *
2906          0015 *
2906          0016 * THIS IS THE INIT ROUTINE
2906          0017 *
2906          0018          ORG IOLOC+16
2910          0019 *
2910 AF 0020 INIS XRA A
2911 D3 03 0021          OUT 3
2913 D3 03 0022          OUT 3
2915 D3 03 0023          OUT 3
2917 3E 40 0024          MVI A,40H
2919 D3 03 0025          OUT 3
291B 3E CE 0026          MVI A,0CEH
291D D3 03 0027          OUT 3
291F 3E 27 0028          MVI A,27H
2921 D3 03 0029          OUT 3
2923 C9 0030          RET
2924          0031 *
2924          0032 *
2924          0033 * THIS CHECKS FOR <@C>, AND SUSPENDS OUTPUT ON <SPACE>
2924          0034 *
2924          0035          ORG IOLOC+64
2940          0036 *
2940 3A FD 29 0037 BUFTM LDA CCBUF
2943 F5 0038          PUSH PSW
2944 AF 0039          XRA A
2945 32 FD 29 0040          STA CCBUF
2948 F1 0041          POP PSW
2949 CD 50 29 0042          CALL CMPR
294C C8 0043          RZ
294D CD DC C0 0044 CHK8 CALL 0C0DCH
2950 FE 03 0045 CMPR CPI 3
2952 C8 0046          RZ
2953 FE 20 0047          CPI 32
2955 C0 0048          RNZ
2956 CD 60 29 0049          CALL IN8
2959 FE 03 0050          CPI 3
295B C9 0051          RET
295C          0052 *
295C          0053 *
295C          0054 * CONSOLE INPUT - <@P> & <@L> ARE HANDLED SPECIALLY

```

295C	0055 *			
295C	0056	ORG	IOLOC+96	
2960	0057 *			
2960 CD DC C0	0058 IN8	CALL	0C0DCH	;GET CHARACTER
2963 CA 60 29	0059	JZ	IN8	
2966 FE 5F	0060	CPI	5FH	
2968 CC 89 29	0061	CZ	MAKES	
296B FE 7F	0062	CPI	7FH	
296D CC 89 29	0063	CZ	MAKES	;CONVERT DEL & USCORE TO CBS
2970 FE 10	0064	CPI	16	;SEE IF @P
2972 CA 7F 29	0065	JZ	CTLPL	
2975 FE 0C	0066	CPI	12	;SEE IF @L
2977 C0	0067	RNZ		;RETURN IF NOT
2978 47	0068	MOV	B,A	
2979 CD C0 29	0069	CALL	OUTPR	;SEND FORMFEED
297C C3 60 29	0070	JMP	IN8	;DISCARD CHARACTER
297F 3A FE 29	0071 CTLPL	LDA	OUTFL	
2982 2F	0072	CMA		
2983 32 FE 29	0073	STA	OUTFL	;@P COMPLIMENTS PRINTER FLAG
2986 C3 60 29	0074	JMP	IN8	;AND DISCARD CHARACTER
2989 3E 08	0075 MAKES	MVI	A,8	
298B C9	0076	RET		
298C	0077 *			
298C	0078 *			
298C	0079 *	OUTPUT CHARACTER - IF A=1, THEN USE DIABLO		
298C	0080 *			
298C	0081	ORG	IOLOC+144	
2990	0082 *			
2990 FE 01	0083 OUT8	CPI	1	
2992 CA C0 29	0084	JZ	OUTPR	;IF A=1 THEN USE PRINTER
2995 78	0085	MOV	A,B	
2996 CD 98 C0	0086	CALL	0C098H	;SEND CHARACTER TO CONSOLE
2999 3A FE 29	0087	LDA	OUTFL	
299C B7	0088	ORA	A	;CHECK PRINTER FLAG
299D 78	0089	MOV	A,B	
299E C8	0090	RZ		;RETURN IF NOT SET
299F C3 C0 29	0091	JMP	OUTPR	;SEND TO PRINTER IF SET
29A2	0092 *			
29A2 32 FF 29	0093 PRCNT	STA	LNCNT	
29A5 78	0094	MOV	A,B	
29A6 C0	0095	RNZ		
29A7 C5	0096	PUSH	B	
29A8 06 0C	0097	MVI	B,12	
29AA CD C0 29	0098	CALL	OUTPR	
29AD C1	0099	POP	B	
29AE 78	0100	MOV	A,B	
29AF C9	0101	RET		
29B0	0102 *			
29B0	0103 *			
29B0	0104 *	DIABLO PRINTER OUTPUT ROUTINE		
29B0	0105 *			
29B0	0106	ORG	IOLOC+192	
29C0	0107 *			
29C0 DB 03	0108 OUTPR	IN	3	
29C2 E6 01	0109	ANI	1	
29C4 CA C0 29	0110	JZ	OUTPR	;WAIT FOR READY FROM USART

29C7 78	0111	MOV	A,B	
29C8 D3 02	0112	OUT	2	;SEND CHARACTER
29CA FE 0A	0113	CPI	10	;SEE IF LINEFEED
29CC CA D9 29	0114	JZ	ITALF	
29CF FE 0C	0115	CPI	12	;SEE IF FORMFEED
29D1 C0	0116	RNZ		;RETURN IF NOT
29D2 3E 38	0117	MVI	A,56	
29D4 32 FF 29	0118	STA	LNCNT	;RESET PAGE COUNTER ON FORMFEED
29D7 78	0119	MOV	A,B	
29D8 C9	0120	RET		
29D9 C5	0121	ITALF	PUSH B	
29DA 06 03	0122	MVI	B,3	;SEND ETX AFTER LF
29DC CD C0 29	0123	CALL	OUTPR	
29DF DB 03	0124	WTACK	IN 3	
29E1 E6 02	0125	ANI	2	
29E3 CA DF 29	0126	JZ	WTACK	;WAIT FOR ACKNOWLEDGE
29E6 DB 02	0127	IN	2	
29E8 E6 7F	0128	ANI	7FH	
29EA FE 06	0129	CPI	06	
29EC CA F5 29	0130	JZ	IACK	
29EF 32 FD 29	0131	STA	CCBUF	
29F2 C3 DF 29	0132	JMP	WTACK	
29F5 C1	0133	IACK	POP B	
29F6 3A FF 29	0134	LDA	LNCNT	
29F9 3D	0135	DCR	A	;ADJUST LINE COUNTER
29FA C3 A2 29	0136	JMP	PRCNT	
29FD	0137 *			
29FD	0138	ORG	IOLOC+253	
29FD	0139 *			
29FD 00	0140	CCBUF	DB 0	;CHAR FROM DIABLO
29FE 00	0141	OUTFL	DB 0	;PRINTER FLAG
29FF 38	0142	LNCNT	DB 56	;LINE COUNTER
2A00	0143 *			

SYMBOL TABLE

BUFT 2940	CCBUF 29FD	CHK8 294D	CMPR 2950	CTLP 297F
IACK 29F5	IN8 2960	INIS 2910	IOLOC 2900	ITALF 29D9
LNCNT 29FF	MAKES 2989	MZOS 2028	OUT8 2990	OUTFL 29FE
OUTPR 29C0	PRCNT 29A2	REENT 2900	WTACK 29DF	

```

2900          0001 * MZOS CENTRONICS I/O SYSTEM
2900          0002 * NEALE BRASSSELL [15-DEC-73]
2900          0003 *
2900          0004 IOLOC EQU 2900H
2900          0005 MZOS EQU 2028H
2900          0006 *
2900          0007 *
2900          0008 * THIS DOES AN INIT, THEN JUMPS TO MZOS
2900          0009 *
2900          0010          ORG IOLOC
2900          0011 *
2900 CD 10 29          0012 REENT CALL INIS
2903 C3 28 20          0013          JMP MZOS
2906          0014 *
2906          0015 *
2906          0016 * THIS IS THE INIT ROUTINE
2906          0017 *
2906          0018          ORG IOLOC+16
2910          0019 *
2910 AF          0020 INIS XRA A
2911 D3 03          0021          OUT 3
2913 D3 03          0022          OUT 3
2915 D3 03          0023          OUT 3
2917 3E 40          0024          MVI A,40H
2919 D3 03          0025          OUT 3
291B 3E CE          0026          MVI A,0CEH
291D D3 03          0027          OUT 3
291F 3E 27          0028          MVI A,27H
2921 D3 03          0029          OUT 3
2923 C9          0030          RET
2924          0031 *
2924          0032 *
2924          0033 * THIS CHECKS FOR <@>, AND SUSPENDS OUTPUT ON <SPACE>
2924          0034 *
2924          0035          ORG IOLOC+64
2940          0036 *
2940 CD DC C0          0037 CHKS CALL 0C0DCH
2943 FE 03          0038          CPI 3
2945 C8          0039          RZ
2946 FE 20          0040          CPI 32
2948 C0          0041          RNZ
2949 CD 60 29          0042          CALL INS
294C FE 03          0043          CPI 3
294E C9          0044          RET
294F          0045 *
294F          0046 *
294F          0047 * CONSOLE INPUT - <@P> & <@L> ARE HANDLED SPECIALLY
294F          0048 *
294F          0049          ORG IOLOC+96
2960          0050 *
2960 CD DC C0          0051 INS CALL 0C0DCH ;GET CHARACTER
2963 CA 60 29          0052          JZ INS
2966 FE 5F          0053          CPI 5FH
2969 CC 89 29          0054          CZ MAKE8

```

```

296B FE 7F          0055      CPI      7FH
296D CC 89 29      0056      CZ       MAKES      ;CONV DEL & USCORE TO BS
2970 FE 10          0057      CPI      16          ;SEE IF @P
2972 CA 7F 29      0058      JZ       CTLP
2975 FE 0C          0059      CPI      12          ;SEE IF @L
2977 C0             0060      RNZ
                ;RETURN IF NOT
2978 47             0061      MOV     B,A
2979 CD C0 29      0062      CALL   OUTPR      ;SEND FORMFEED
297C C3 60 29      0063      JMP     IN8        ;DISCARD CHARACTER
297F 3A FE 29      0064      CTLP   LDA      OUTFL
2982 2F             0065      CMA
2983 32 FE 29      0066      STA      OUTFL      ;@P COMPLIMENTS OUTPUT FLAG
2986 C3 60 29      0067      JMP     IN8        ;AND DISCARD CHARACTER
2989 3E 08          0068      MAKES  MVI     A,8
298B C9             0069      RET
298C               0070 *
298C               0071 *
298C               0072 * OUTPUT CHARACTER - IF A=1, THEN USE CENTRONICS
298C               0073 *
298C               0074      ORG     IOLOC+144
2990               0075 *
2990 FE 01          0076      OUT8   CPI      1
2992 CA C0 29      0077      JZ      OUTPR      ;IF A=1 THEN USE PRINTER
2995 78             0078      MOV     A,B
2996 CD 98 C0      0079      CALL   @C098H     ;SEND CHARACTER TO CONSOLE
2999 3A FE 29      0080      LDA      OUTFL
299C B7             0081      ORA      A
                ;CHECK PRINTER FLAG
299D 78             0082      MOV     A,B
299E C8             0083      RZ
                ;RETURN IF NOT SET
299F C3 C0 29      0084      JMP     OUTPR      ;SEND TO PRINTER IF SET
29A2               0085 *
29A2               0086 *
29A2               0087 * CENTRONICS PRINTER OUTPUT ROUTINE
29A2               0088 *
29A2               0089      ORG     IOLOC+192
29C0               0090 *
29C0 DB 01          0091      OUTPR  IN8
29C2 E6 01          0092      ANI     1
29C4 C2 C0 29      0093      JNZ    OUTPR      ;WAIT TILL NOT BUSY
29C7 78             0094      MOV     A,B
29C8 F6 80          0095      ORI     128
                ;SET STROBE
29CA D3 01          0096      OUT
29CC E6 7F          0097      ANI     127
                ;CLEAR STROBE
29CE D3 01          0098      OUT
29D0 F6 80          0099      ORI     128
                ;SET STROBE (NOW WE'VE PULSED IT)
29D2 D3 01          0100      OUT
29D4 78             0101      MOV     A,B
29D5 FE 0A          0102      CPI     10
                ;SEE IF LINEFEED
29D7 CA E4 29      0103      JZ     ITALF
29DA FE 0C          0104      CPI     12
                ;SEE IF FORMFEED
29DC C0             0105      RNZ
                ;RETURN IF NOT
29DD 3E 38          0106      MVI     A,56
29DF 32 FF 29      0107      STA      LNCNT     ;RESET PAGE COUNTER ON FORMFEED
29E2 78             0108      MOV     A,B
29E3 C9             0109      RET
29E4 3A FF 29      0110      ITALF  LDA      LNCNT
                ;ON LINEFEED, ADJUST LINE-COUNTER

```

```

29E7 3D          0111      DCR   A
29E8 32 FF 29   0112      STA  LNCNT
29EB 78          0113      MOV  A,B
29EC C0          0114      RNZ          ;RETURN IF LESS THAN A PAGE
29ED C5          0115      PUSH B
29EE 06 0C      0116      MVI  B,12
29F0 CD C0 29   0117      CALL OUTPR  ;IF FULL PAGE, SEND FORMFEED
29F3 C1          0118      POP  B
29F4 78          0119      MOV  A,B
29F5 C9          0120      RET          ;AND RETURN
29F6             0121 *
29F6             0122      ORG  IOLOC+254
29FE             0123 *
29FE 00          0124 OUTFL DB  0      ;PRINTER FLAG
29FF 38          0125 LNCNT DB  56     ;LINE COUNTER
2A00             0126 *

```

SYMBOL TABLE

```

CHK8  2940      CTLP  297F      IN8   2960      INI8  2910      IOLOC 2900
ITALF 29E4      LNCNT 29FF      MAKES 2989     MZOS  2028     OUT8  2990
OUTFL  29FE     OUTPR  29C0     REENT 2900

```

```

2900          0001 * MZOS SERIAL I/O SYSTEM
2900          0002 * NEALE BRASSELL [15-DEC-78]
2900          0003 *
2900          0004 IOLOC EQU 2900H
2900          0005 MZOS EQU 2028H
2900          0006 *
2900          0007 *
2900          0008 * THIS DOES AN INIT, THEN JUMPS TO MZOS
2900          0009 *
2900          0010          ORG IOLOC
2900          0011 *
2900 CD 10 29          0012 REENT CALL IN8
2903 C3 28 20          0013      JMP MZOS
2906          0014 *
2906          0015 *
2906          0016 * THIS IS THE INIT ROUTINE
2906          0017 *
2906          0018          ORG IOLOC+16
2910          0019 *
2910 AF          0020 IN8 XRA A
2911 D3 03          0021      OUT 3
2913 D3 03          0022      OUT 3
2915 D3 03          0023      OUT 3
2917 3E 40          0024      MVI A,40H
2919 D3 03          0025      OUT 3
291B 3E CE          0026      MVI A,0CEH
291D D3 03          0027      OUT 3
291F 3E 27          0028      MVI A,27H
2921 D3 03          0029      OUT 3
2923 C9          0030      RET
2924          0031 *
2924          0032 *
2924          0033 * THIS CHECKS FOR <@C>, AND SUSPENDS OUTPUT ON <SPACE>
2924          0034 *
2924          0035          ORG IOLOC+64
2940          0036 *
2940 CD DC C0          0037 CHK8 CALL 0C0DCH
2943 FE 03          0038 CMPR CPI 3
2945 C8          0039      RZ
2946 FE 20          0040      CPI 32
2948 C0          0041      RNZ
2949 CD 60 29          0042      CALL IN8
294C FE 03          0043      CPI 3
294E C9          0044      RET
294F          0045 *
294F          0046 *
294F          0047 * CONSOLE INPUT - <@P> & <@L> ARE HANDLED SPECIALLY
294F          0048 *
294F          0049          ORG IOLOC+96
2960          0050 *
2960 CD DC C0          0051 IN8 CALL 0C0DCH ;GET CHARACTER
2963 CA 60 29          0052      JZ IN8
2966 FE 5F          0053      CPI 5FH
2968 CC 89 29          0054      CZ MAKE8

```

296B FE 7F	0055	CPI	7FH	
296D CC 89 29	0056	CZ	MAKES	;CONVERT DEL & USCORE TO BS
2970 FE 10	0057	CPI	16	;SEE IF @P
2972 CA 7F 29	0058	JZ	CTLP	
2975 FE 0C	0059	CPI	12	;SEE IF @L
2977 C0	0060	RNZ		;RETURN IF NOT
2978 47	0061	MOV	B,A	
2979 CD C0 29	0062	CALL	OUTPR	;SEND FORMFEED
297C C3 60 29	0063	JMP	IN8	;DISCARD CHARACTER
297F 3A FE 29	0064	CTLP LDA	OUTFL	
2982 2F	0065	CMA		
2983 32 FE 29	0066	STA	OUTFL	;@P COMPLIMENTS PRINTER FLAG
2986 C3 60 29	0067	JMP	IN8	;AND DISCARD CHARACTER
2989 3E 08	0068	MAKES MVI	A,8	
298B C9	0069	RET		
298C	0070	*		
298C	0071	*		
298C	0072	*		OUTPUT CHARACTER - IF A=1, THEN USE PRINTER
298C	0073	*		
298C	0074	ORG	IOLOC+144	
2990	0075	*		
2990 FE 01	0076	OUT8 CPI	1	
2992 CA C0 29	0077	JZ	OUTPR	;IF A=1 THEN USE PRINTER
2995 78	0078	MOV	A,B	
2996 CD 98 C0	0079	CALL	0C098H	;SEND CHARACTER TO CONSOLE
2999 3A FE 29	0080	LDA	OUTFL	
299C B7	0081	ORA	A	;CHECK PRINTER FLAG
299D 78	0082	MOV	A,B	
299E C8	0083	RZ		;RETURN IF NOT SET
299F C3 C0 29	0084	JMP	OUTPR	;SEND TO PRINTER IF SET
29A2	0085	*		
29A2	0086	*		
29A2	0087	*		SERIAL PRINTER OUTPUT ROUTINE
29A2	0088	*		(TTY, DECWRITER, T1810, ETC.)
29A2	0089	*		
29A2	0090	ORG	IOLOC+192	
29C0	0091	*		
29C0 DB 03	0092	OUTPR IN	3	
29C2 E6 01	0093	ANI	1	
29C4 CA C0 29	0094	JZ	OUTPR	;WAIT FOR READY FROM USART
29C7 78	0095	MOV	A,B	
29C8 D3 02	0096	OUT	2	;SEND CHARACTER
29CA FE 0A	0097	CPI	10	;SEE IF LINEFEED
29CC CA D9 29	0098	JZ	ITALF	
29CF FE 0C	0099	CPI	12	;SEE IF FORMFEED
29D1 C0	0100	RNZ		;RETURN IF NOT
29D2 3E 38	0101	MVI	A,56	
29D4 32 FF 29	0102	STA	LNCNT	;RESET PAGE COUNTER ON FORMFEED
29D7 78	0103	MOV	A,B	
29D8 C9	0104	RET		
29D9 3A FF 29	0105	ITALF LDA	LNCNT	
29DC 3D	0106	DCR	A	;ADJUST LINE COUNTER
29DD 32 FF 29	0107	STA	LNCNT	
29E0 78	0108	MOV	A,B	
29E1 C0	0109	RNZ		
29E2 C5	0110	PUSH	B	

