MSCSI™

MSCSI:
Software Interface Package
OPERATOR MANUAL

MSCSI™

MSCSI:
Software Interface Utility Package
OPERATOR MANUAL

Revision 1.0.0 09/01/89

MISOSYS, Inc
PO Box 239
Sterling, VA 22170-0239
703-450-4181

Software License Agreement

MISOSYS, Inc., authorizes you to use this software on only one computer at a time. The purchaser is entitled to make as many working copies of this disk as is needed for his or her personal use.

Warranty

MISOSYS, Inc., warrants the physical diskette and physical documentation to be free of defects in materials and workmanship for a period of 90 days from the date of purchase. Upon notification of defects in material or workmanship within the warranty period, MISOSYS, Inc., will replace the defective documentation or diskette.

MISOSYS, Inc disclaims all other warranties, expressed or implied, including but not limited to any implied warranty of merchantability and/or fitness for particular purpose. Under no circumstances shall MISOSYS, Inc be liable for any loss of profit or any other damage, including but not limited to special, incidental, exemplary, consequential or other damages.

MSCSI: Software Interface Package

Table of Contents

General Information

MSCSI is a software interface package that provides a hard disk driver, formatter, large-file archive and restore utilities, and other utilities, which support certain hard drive units manufactured and/or assembled by MISOSYS.

Specifically, MSCSI supports up to two drive units interfaced via the MISOSYS SCSI host adaptor coupled with an Adaptec 4000 or a Xebec S1420 hard disk controller (or equivalent). The driver supports drives of up to 1024 cylinders with up to eight (8) heads.

Because drive configurations may exceed the maximum limits of a single logical drive afforded by the DOS, the driver supports drive partitioning by both head and cylinder.

The MSCSI package is provided on a 40-track double density data diskette. It has software which supports the MISOSYS hard drive used with one of the following two operating systems: LDOS Version 5.3 or LS-DOS 6.3.

The files on the accompanying DATA diskette may be easily copied to your working SYSTEM disk by means of the DOS COPY utility. There may be a file named "README/TXT" on the disk. If so, this file will contain important information which may not appear in this printed documentation. You should read this file by issuing the command:

```
LIST README
```

The following files are included on the software diskette:

ARCHIVE/CMD	Utility to make backups of files from your hard disk to multiple floppy diskettes.
ERROR/CMD	A utility program which can be used to display a descriptive message for the "last error encountered" recognized by the MSCSI hard disk driver.
FSCSI/CMD	Utility to perform the low level formatting of the hard drive.

HD20INIT/JCL	Job Control Language file used to install a 20 megabyte hard drive.
HD40INIT/JCL	Job Control Language file used to install a 40 megabyte hard drive.
HDCHECK/CMD	A utility program used to perform testing of all sectors on a hard drive.
JSTICK/FLT	A filter program for the optional joystick which converts the joystick operation to keystrokes.
MALARM/CMD	A utility program used in connection with the hardware clock option to operate the alarm functions.
MCLOCK/CMD	A utility program used in conjunction with the hardware clock option to set the clock and read clock values into the system storage areas.
MSCPARK/CMD	Utility to use when powering down your hard drive to park the drive's heads into a special "parking zone".
MSCSI/DCT	A driver used to interface the hard drive unit to the operating system.
MSCSIF/CMD	A utility to provide high-level formatting of the hard drive.
README/TXT	A file which contains information over and above what is contained in this operator's manual.
RESTORE/CMD	A utility to place archived files back onto a hard drive.
SD/CMD	A utility program which either sets up an existing SubDISK file as a drive, or disables an active SubDISK.

SDFORM/CMD	A SubDISK formatter utility program which initially creates a SubDISK;
SETJS/CMD	A utility to alter the optional joystick generated keycodes after the joystick filter is installed.
SWAP/CMD	A utility program to switch logical drive numbers while running the installation Job Control Language files.

About your hard drive

As this is the operator's manual associated with the installation and use of the software interface, details concerning the physical drive unit are provided in a separate hardware operator's manual. But just to get you started, some brief instructions are included here.

Your hard drive kit assembled by MISOSYS is equipped with either a 20 or 40 Megabyte hard drive. Unless specified elsewhere, drives are ST406-type (MFM recording). It uses a hard disk controller (HDC) with a Small Computer Systems Interface (SCSI) on the host side to interface to the drive. A proprietary host adaptor (H/A) connects the SCSI interface of the HDC to the expansion bus of the TRS-80. The hard drive unit has a rear panel which includes the following:

1. AC Power receptacle: left bottom
2. on/off power switch: left rear top
3. fuse socket: left rear middle
4. 50-pin SCSI female bus connector: right top
5. optional DB9 joystick port connector: middle
6. fan air flow screen: right bottom

The drive can be powered from either 115V or 230V AC. This is optioned by means of a plug inside the unit mounted on the power supply. **Note that the unit as provided by MISOSYS is optioned for 115 Volt AC.** Consult the hardware operators manual for any conversion required.

Connecting your hard drive

A 50-pin ribbon cable with a SCSI male connector on one end and an edgecard connector on the other end connects the hard drive to your machine. The SCSI connector looks like a standard Centronics printer connector, but it is larger as it contains 50-pins. This connector can only be oriented one way on the hard drive. The other end containing the 50-pin edgecard connector should be plugged into the 50-pin expansion port of your computer. On a desktop Model 4 or 4D, this port is on the bottom rear middle of the computer. With the power turned off, turn your computer on

end and plug in the edgecard connector with the ribbon cable exiting towards the rear. One end of the ribbon cable has a blue stripe running down its side. This side will be oriented towards the right of your computer as you are looking at the screen. On a Model 4P, the ribbon cable is facing downward. Restore your computer to its proper position.

Plug the supplied AC power cable into the hard drive power receptacle and the other end into an AC power socket. Turn on the hard drive power using the switch located on the rear of the hard drive cabinet. Turn on your computer. Proceed to the sections covering software installation.

Procedures to interface a hard drive

Interfacing a hard drive (or drives) is not a difficult task. The job can be divided up into a few separate operations. An outline of these steps is:

1) Read the section on partitioning a hard drive so that you can decide on how you want to divide up the storage space on your drive. Once you decide, keep a written record of your decision.

2) Invoke the low-level formatter, FSCSI, to apply low-level format information to the hard disk drive. Note that this formats the entire hard drive specified. Such low-level formatting is initially done by MISOSYS prior to shipment of your hard drive pre-assembled kit; you will rarely have to perform this step.

3) Invoke the hard disk driver for each partition you have decided upon. The section, "Invoking the Hard Disk Driver" contains complete instructions on this operation.

4) Use the MSCSIF command to add the operating system directory information to each partition established in step 3.

After following the preceding four steps, your drive is ready for use. Note that steps 3 and 4 can be invoked via the HDxxINIT/JCL Job Control Language file. Each JCL file partitions the drive into four partitions with the first partition being the system drive. It would be a good idea to read this entire manual to gain familiarity with the archival, restoral utilities, and the SubDISK facility.

Hard drive partitioning

The term "partitioning" refers to segmenting a hard drive into two or more logical drives. The integrator of a hard disk usually has to consider some form of hard disk partitioning. Why is this to be considered? A hard disk has a minimum of five megabytes of storage space. The demand for storage never abates; thus, ten megabyte, twenty megabyte, and higher capacities are being integrated into the microcomputer environment.

Both LDOS and LS-DOS have limitations on the total size of a storage device that is addressable as a single volume. These are limitations stemming from the size of the directory. A device is limited to a maximum of 256 sectors per logical cylinder, and 203 logical cylinders [406 physical

cylinders]. Given a standard sector size of 256 bytes, the DOS can address 13.3 megabytes total on a single volume. If the physical drive exceeds this capacity, then it must be divided into more than one logical drive in order to address its total capacity.

The DOS also limits the number of files per logical drive to 256 [The DOS always reserves two directory slots taken up by the BOOT/SYS and DIR/SYS files, and reserves fourteen additional directory slots for the /SYS files on "system" drives. Although data base applications may find the most practical arrangement is a single volume, the typical use of even a five megabyte drive will find the file slots filled before all of the space is allocated - thus space is wasted.

The SD sub-disk partitioning utility provided as part of this package may also be used to emulate multiple logical floppy disk drives by means of disk files on your hard drive and a small driver program module to provide a great deal more sub-partitions, and thus directory slots. This is useful for creating mirror image backups from a drive with larger storage capacity to a drive with lesser storage. It is also useful for partitioning a large disk volume (i.e. hard disk) into smaller regions. This effectively overcomes the limitation of 256 directory entries per "disk".

Once the decision is made to divide a drive, the question arises as to how to go about such a division. There are three methods of partitioning. One is to divide the drive by cylinder. For example, Take a 306 cylinder, four head, 10 megabyte drive. This can be divided into two drives with the first logical drive using cylinders 1-153 while the second uses cylinders 154-306. The DOS actually uses logical cylinder numbers 0-152 for both partitions and the hard disk driver must recognize that it needs to translate the 0-152 for the second partition into the range 154-306. Obviously, one can divide up the drive into partitions smaller than five megabytes.

A second method is to divide the drive so that all of the cylinders are included in a single logical volume, but volumes use different heads. Thus, the previously mentioned drive could be divided into two, three, or four logical drives.

A third method would be to partition by both head and cylinder. The MSCSI driver can accept both head and cylinder specifications for a logical partition.

There are advantages and disadvantages to each method. First, if you are demanding the fastest data transfer rate on sequential read/write operations, it is usually advantageous to address as much capacity in a single cylinder prior to having to step the drive to a new cylinder. This means that you would lean towards divisions by cylinder. However, if you are alternately selecting different partitions, the drive must be stepped a great distance to get to each partition. Of course, if the drive physically has more than 406 cylinders, it must be partitioned by cylinder to address the higher numbered cylinders.

Partitioning by head provides less sectors per physical cylinder; however, since hard drives today usually use very fast buffered seek, the stepping time to advance a track is minimal.

Before you set out to assign your drive, you should consider the uses of your hard drive. How many file slots do you need? How many distinct logical drives do you need? Choose how you will partition the drive. Keep written notes as to your scheme.

Make note that the sub-disk partitioning facility will provide additional means to divide up your hard drive into much smaller pieces. So it may be wise to consider installing the driver with large partitions, using SD to create smaller disk images.

The Job Control Language files supplied with this interface, which automatically install the software, partition the drive by cylinder; all heads are used for each partition.

Simplified Software Installation

If you are prone to avoid delving into instructions for manual installation of the software interface, rest assured that MISOSYS has not ignored you. There are two automated installation procedures provided on the software interface disk. These procedures are implemented as Job Control Language files: one is HD20INIT/JCL for the 20 megabyte drive unit; the other is HD40INIT/JCL for the 40 megabyte drive unit. Each will assume that you have a two-floppy system, will divide the hard drive into four partitions, will make the first partition the "SYSTEM" drive, will BACKUP your BOOT floppy to that system partition, will make the hard drive the SYSTEM drive, and will number the hard drive partitions as :0, :1, :2, and :3, and will number your two existing floppy drives as :4 and :5. All you need do is to invoke the JCL file with the command,

```
DO HDXXINIT
```

then manually type one command at the conclusion of the JCL automated installation processing. Of course the "XX" would be entered either as "20" or "40", whichever is applicable to your hard drive unit.

Caution: There may be additional instructions for this procedure described in the README/TXT file located on the software installation disk.

Low-level formatting the Hard Disk Drive

The first procedure needed for a new hard drive is the application of low-level formatting. This procedure is initiated by the FSCSI utility, and is where the characteristics of the drive, such as numbers of cylinders, numbers of heads, etc.) are input. Note that this operation is generally performed only once, will format an entire hard drive, and is performed at MISOSYS prior to shipping your drive kit. This saves you from having to perform the operation. If you change drive units, or are providing your own drive unit, you will need to initiate this procedure.

Since formatting the entire drive is a rather drastic procedure, certain safeguards have been built into the formatter. One safeguard is that it will not carry out its work if it finds that the MSCSI hard disk driver is installed and "connected" to the drive you wish to low-level format. The low-level formatter is invoked via the command:

FSCSI (Drive="dx")
FSCSI (Drive=d)

d	Specifies the **physical** drive number <1 or 2>; defaults to 1.
x=f|r	Specifies the drive as being a fixed device (f) or a removable device (r); defaults to "f".
Abbrev	Drive=D

When control is passed to the low-level formatter program, FSCSI, it will display the following message:

```
FSCSI Low Level Formatter - Version x.y.z -
for ********
Copyright (C) 1989 by MISOSYS, Inc., All
rights reserved
```

The "x.y.z" field references the version number of the software; the "********" field references the connected controller type: Xebec 1420 or Adaptec 4000.

The first item of information pertinent to a hard disk drive is the drive's physical address. The MSCSI driver package supports a maximum of two drives. These are addressed as one (1) and two (2) which correspond to a device selection jumper assignment physically on the disk drive. The first drive will have an address of one. The physical drive address can be entered as a parameter on the command line; however, if you omit it there, you will be asked for input. The address is entered in response to the prompt:

```
Enter drive address <1,2> and F/R [1;F] -----
------>
```

The "F/R" refers to FIXED and REMOVABLE drive packs. As can be noted, the default used is ADDRESS 1 and FIXED. If you are addressing drive address 2 on a fixed pack drive, you only need enter "2" - the "F" default will still be used. If you need to specify the drive as REMOVABLE, then you must enter the drive address. When you make the entry, the characters are entered consecutively without any space, comma, or semi-colon. For instance, to select a removable drive at address one, enter "1R" or "1r".

Before the disk controller can function with any hard drive, information on the number of cylinders, number of heads, and step rate option of the drive(s) must be known. Since this information is common to all partitions of a drive connected to the controller, you only need enter this information once. This is done during the invocation of FSCSI. Subsequent invocations of the hard disk driver, MSCSI, will use the values saved from this format operation.

Many of the prompts support a default value whereupon if you depress only the <ENTER> key without a value, the default will be chosen. The default value is displayed within square brackets "[]" when the prompt message is displayed. If you want to select this default value, it is necessary to depress just the <ENTER> key. If you depress the <BREAK> key in response to any prompt, the program will terminate and display the following message:

```
Manual abort - job terminated!
```

Because hard drive must be within certain boundaries, the formatter program will NOT permit you to do enter an invalid value. When it detects a value out of bounds, it will reprompt you to enter a valid value.

The FSCSI program first checks to see if the MSCSI driver is already installed and connected to the drive you specified. This is a cautionary safeguard so that you cannot write over your valuable data with the low-level format once you have installed the driver. If the driver is located and "enabled", meaning already in use, the error message:

```
Can't low-level format an active drive!
```

will be displayed and the program will terminate.

Controller Parameter Prompts

Once you make the formatter aware of which drive to format, it checks to see if the drive has been previously formatted (with a compatible controller). If it finds proper information stored on the drive, the obtainable current characteristics may be displayed in the form:

```
Current: Cylinders [cccc], Heads[h], Step
rate [s]
Do you want to revise drive parameters <Y,N>
?
```

If you respond with "n" OR "N", the queries for the controller parameters will be bypassed.

The first prompt associated with the controller parameters is:

```
Enter the STEP RATE OPTION <0-3> [2] ------->
```

The option numbers correspond to the following step rates:

0 = 3 milliseconds
1 = 30 microseconds
2 = 15 microseconds
3 = 15 microseconds

Observe that the step rate option default is "2". This is a typical value for modern day drives as current drives use a fast buffered-seek stepping. If the step rate option of your drive is "2", then you only need depress <ENTER>.

On the other hand, if you noted that information supplied with your drive unit specifies a different rate, then adjust the entry accordingly.

The second prompt associated with the controller parameters is:

```
Enter the total CYLINDER count [0615] ------>
```

This figure is the total number of tracks on a single surface. The default value is 615 cylinders as shown in the prompt. This value is for the 20 megabyte drive. Another value that is generally used is 820 cylinders for the 40 megabyte drive. If your drive has 820 cylinders, then enter the digits "820" followed by <ENTER>. The maximum number of cylinders supported in this driver is 1024 cylinders.

The next item of information which is needed for an Adaptec controller is the cylinder landing zone for power down. Enter the number in response to the prompt:

```
Enter the power down landing zone [0670] --->
```

This figure may be used to initialize the landing zone used by the MSCPARK program. The Seagate ST225 and ST251 twenty and forty megabyte drives identified above have a landing zone of cylinder 670 and 820 respectively. Use the MSCPARK program when you are powering down your drive.

The last prompt associated with the controller parameters is:

```
Enter the total number of HEADS <1-8> [4] -->
```

The twenty megabyte drive identified above has four heads; the forty megabyte drive identified above has six heads.

Information supplied with your drive will advise you of any changes. There are two heads per platter; therefore, a one-platter drive has two heads, a two-platter drive has four heads, and a three-platter drive has six heads. The maximum number of heads supported in this package is eight heads. If your drive has four heads (two platters), you only need to depress <ENTER>; otherwise, enter the digit specifying the number of heads followed by <ENTER>.

Once you have entered the data items mentioned above, FSCSI provides one more opportunity to abort. It displays the following message:

```
This operation will format your entire hard
drive
Are you sure this is what you want <Y,N> >> >
```

If you do not respond with "Y" or "y", the program will abort. Otherwise, the message,

```
Formatting... Please be patient.
```

will be displayed while the drive is being formatted. Upon a successful completion of the low-level format operation, the message,

```
Low level format now applied
```

will be displayed.

Invoking the Hard Drive Disk Driver

The hard disk driver associates a hard drive or partition of a hard drive to a logical drive slot in the operating system. The drive slot requested must be in a "disabled" condition. Any partitioning of the hard drive is specified via responses to the prompts that are displayed during the driver initialization. The driver is invoked via the command:

SYSTEM (DRIVE=d,DISABLE,DRIVER="MSCSI")

d	Specifies the logical drive number <1-7>.

Each partition of a hard drive must be identified via a separate invocation of the MSCSI program. When control is passed to the driver initialization program, MSCSI, it will display the following message:

```
SCSI Hard Disk Driver - Version x.y.z - for
********
Copyright (C) 1989 by MISOSYS, Inc., All
rights reserved
```

The "x.y.z" field references the version number of the software; the "********" field references the connected controller type: Xebec 1420 or Adaptec 4000.

Before the disk controller can function with any hard drive, information on the number of cylinders, number of heads, and step rate option of the drive(s) must be known. Since this information is common to all partitions of a drive, you only need enter this information once. This is done during the low-level formatting and stored on the drive by the controller. Thus, once the physical drive select address is known, the driver first checks the specified drive to retrieve the characteristics. If it finds the drive without a proper low-level format, it displays the following error message:

```
Can't install without low-level format first!
```

Prompts request information concerning the number of heads in a partition, the starting head number, and the number of cylinders in a partition. Many

of the prompts support a default value whereupon if you depress only the <ENTER> key without a value, the default will be chosen. The default value is displayed within square brackets "[]" when the prompt message is displayed. If you want to select this default value, it is necessary to depress just the <ENTER> key. If you depress the <BREAK> key in response to any prompt, the program will terminate and display the following message:

```
Manual abort - job terminated!
```

Because hard drive partitioning gives you the capability to select heads and cylinders, there is the possibility of entering a specification that conflicts with logical partitions already assigned. The driver initialization program will NOT permit you to do this. When it detects such a conflicting request, it will reprompt you to enter a valid value.

The MSCSI program checks to see if the logical drive slot you specified in the "SYSTEM ..." command is available for use. This guards against inadvertent disabling of an active drive. A particular logical drive slot is available only if the slot has been "disabled". The slot will be disabled either if you had entered the "DISABLE" parameter or the slot was previously disabled. If the slot is "enabled", meaning already in use, the error message:

```
Requested drive slot already in use!
```

will be displayed and the program will terminate.

Data I/O Error reporting

In order to keep the size of the hard disk driver to a minimum, all errors detected by the hard disk controller (HDC) are converted to "Unknown error code". The actual HDC error recognized may be displayed by invoking the program, ERROR. This will then display a descriptive error message for the "last error encountered".

Logical Drive Parameters

The first item of information pertinent to a logical drive is the drive's address. The MSCSI driver software supports up to two drives. These are addressed as one (1) and two (2). The first drive will have an address of one. The address is entered in response to the prompt:

```
Enter drive address <1,2> and F/R [1;F] ---->
```

The "F/R" refers to FIXED and REMOVABLE drive packs. As can be noted, the default used is ADDRESS 1 and FIXED. If you are addressing drive address 2 on a fixed disk drive, you only need enter "2" - the "F" default will still be used. If you need to specify the drive as REMOVABLE, then you must enter the drive address. When you make the entry, the characters are entered consecutively without any space, comma, or semi-colon. For instance, to select a removable drive at address one, enter "1R" or "1r".

The MSCSI driver automatically starts assigning space starting from the first available cylinder of the drive. Subsequent invocations of the driver determine what tracks of what heads are in use by examining the data stored in the system's tables associated with each drive slot. If you are working with a huge drive, you may want to restrict the driver from using some portion of the drive. This can be accomplished by inhibiting a range of tracks on one or more heads. Your information is input in response to prompts. The driver will display,

```
Avoid using any drive region <Y,N> ? [N] --->
```

As noted, the default is "N" which implies that you do not want to make any specifications to restrict the driver from using any portion of the drive. If you do wish to make such restrictions, enter a "Y" or "y".

You first enter the number of the head where you wish to inhibit a range of tracks. Do this in response to the query,

```
Enter physical head number <1-n> ---------->
```

Then enter the range of tracks on that head in response to the query,

```
Enter physical track number <0001-tttt> ---->
```

Enter both the starting track number and the ending track number separating both numbers with a hyphen (as shown). The message displays the minimum (0001) and maximum (tttt) values available.

After completing the entry, the driver returns to the "*Avoid using...*" prompt. Continue to manually enter as many ranges on as many heads as you so

desire. If you have a significant number of entries to make, consider creating a job control language file to install the driver.

The next item of information needed is the number of heads assigned to the logical drive. Remember, you may partition a hard drive by head with one or more heads assigned to a logical drive. This entry is entered in response to the prompt message:

```
Enter partition's number of heads <1-n> ---->
```

Now in this message, the "n" will actually be a number that is indicative of the number of heads left unassigned. For example, the first selection in a four-head drive will display "<1-4>". If only one head remains to be assigned, the message will NOT be displayed but that head will automatically be selected for assignment. Since a drive may be partitioned by cylinders as well as by heads, a head will be shown as available provided at least one cylinder is unassigned.

After you have made your selection as to the head quantity, you then can specify the starting head number. An example will illustrate this concept. Suppose you are selecting two heads of a four-head drive. You can use any of the following heads: 1&2, 2&3, or 3&4. The starting head number will be 1, 2, or 3 respectively. Make your selection and respond to the prompt message:

```
Heads already in use <.-.-.-.>
Enter starting head -->
```

In this message, each dot "." corresponds to a head number 1-n. If any heads are already totally assigned, they will appear in the "<.-.-.-.>" field. For example, if the first head was in use, the field will be displayed as "<1-.-.-.>". The number of dot positions displayed in the message corresponds to the total number of heads for the drive. If the number of heads remaining is equal to the number of heads requested for this partition, then the prompt will NOT be displayed and the obvious selection will be automatically made.

If you enter a request that conflicts with an existing assignment (i.e. request a starting head number that is already assigned or request two heads with a usable starting head but the next head is already assigned), the program will display an error message of the form:

```
Heads requested conflict with heads in-use!
```

and prompt for another entry to the "starting head" message.

The last item of information needed to convey to the initialization program is the number of cylinders to assign to the logical partition. This number is entered in response to the prompt:

```
Enter partition's number of cylinders [xxx] >
```

In this message, the "xxx" represents the actual number of physical cylinders available using the heads you have requested. This will be the default used if you depress just the <ENTER> key. In no case can you choose a number which exceeds 406 cylinders. The program will automatically choose the first available cylinder as the starting cylinder for this logical drive.

At the conclusion of the entries, the driver checks to see if the drive partition just identified has ben high-level formatted. If it has not had the system information applied (see next section), the following message will be displayed.

```
Note: Drive appears to be unformatted
```

This is just an informative message; there is no need for alarm, unless, of course, you know that the system information has been applied.

Information Retention

It is strongly recommended that you record on paper, the responses to each prompt in case the procedure has to be repeated. This information may be put on a label and affixed to the drive. It is important data that will be useful for the day that you have to perform the identical procedure when you reconfigure your system. Remember, in order to properly recover files from a partitioned drive that has been reconfigured, it is essential that it be reconfigured in exactly the same manner.

If your configuration uses a removable drive pack, then you are strongly cautioned to always partition each removable cartridge in exactly the same configuration. If you ignore this recommendation, you may destroy the files when a cartridge is changed.

Important

Once you configure all partitions of a disk drive, you must FORMAT each partition (see the next section) to add the directory information. This must be done before you invoke any other command or program - especially the DOS command, DEVICE. If your disk drive was previously high-level formatted in the IDENTICAL configuration, you may bypass the drive format operation. Also, so that your "configuration" will be available when you reBOOT your system, you should SYSGEN [consult your DOS manual for SYSTEM (SYSGEN)]. Also see the section, "Using the Hard Drive as the SYSTEM Drive".

Formatting the Hard Drive

Before you can use the hard drive, it must be formatted and written with system information (i.e. the BOOT/SYS and DIR/SYS files - the DIR/SYS file contains the directory for the logical drive). Formatting is performed via the command, MSCSIF. This operation will format only the logical drive partition connected to the specified drive slot. Once the drive is formatted, all information that was previously kept in that partition of the drive is eliminated.

The MSCSIF command is entered as:

MSCSIF :d (Name="diskname",Mpw="password", Verify=N,Dir=c,NOSTOP)

Name=	Specifies the name to be applied to the disk drive. This name appears in DIR, FREE, and DEVICE commands
Mpw	This name provides the master password for the drive; it is recommended that you choose a password other than "password".
Verify=N	To ensure overall integrity of each sector on the disk, the formatter will verify the readability of each sector. Any found to be not readable, will be locked out from use. Verification takes a great deal of time. A quick re-format may be invoked by specifying V=N.

continued

	continuation
Dir=	Normally the formatter will place the directory for the drive at the approximate middle cylinder. If you want to force the selection of a directory cylinder of your choosing, enter it with this parameter as "D=ccc", where "ccc" is the cylinder of your choice.
NOSTOP	Normally, the formatter will check the current state of the drive for existing directory information. This can produce unexpected prompts. If you are going to invoke the formatter from Job Control Language and you want to inhibit those tests, specify the NOSTOP parameter.
Abbrev:	Name=N,Mpw=M,Verify=V, Dir=D

The "packname" should be representative and distinct It is recommended that you affix a number that is associated with the number of the partition. It is also recommended that you choose a password that is something other than "PASSWORD". This precaution will be your saving grace when you inadvertently enter a DOS command that completes undisturbed if the disk's master password was "PASSWORD". The JCL procedure will use names of "MSCHARD1", "MSCHARD2", etc., and password of "HARDDISK". Repeat this high-level formatting operation for each partition of the drive you installed.

If you have a second hard drive, repeat the low-level formatting, if required, driver installation, and high-level formatting steps for that drive. Thereafter, you should not have to re-format your drives.

Checking the Hard Drive - Using HDCHECK

The HDCHECK utility is designed to check whether each sector of the hard drive is readable (and optionally writeable). Where an error is detected, the granule containing that sector can be optionally locked out for further use. Since each sector of the hard drive is verified during the high-level formatting, HDCHECK can be used after formatting and applying the system information to all partitions. It is not necessary to run HDCHECK; it is provided only as a tool to test your drive at some future point in time. HDCHECK is invoked via the command:

HDCHECK :d (Lock,Query=N,Write)

d	Specifies the drive number to be checked.
Lock	Unless this parameter is entered, HDCHECK will only advise you of errors it finds. Specify the LOCK parameter to keep the system from using these bad sections of the drive.
Query=N	If you specify LOCK and a bad sector is found, you will be prompted prior to locking out the granule. If you do not want the prompts, then specify Q=N.
Write	HDCHECK will check a drive by reading every sector (except for the directory which was already checked by FORMAT). Reading will usually discover all errors that may exist. A greater stress test is to write all sectors as well. HDCHECK uses a non-destructive write by first reading then re-writing the data that was read. The WRITE check will take longer.
Abbrev:	L=Lock, Q=Query, W=Write

It is suggested that you use the WRITE and LOCK parameters. For example, a suitable command invocation is:

```
HDCHECK :4 (LOCK,WRITE)
```

The message, *Checking Cylinder: ccc, Sector: sss* is displayed as each sector is being checked. HDCHECK will display either of the messages, *Lockout <Y,N,C> ?* or *Automatically locked* depending on whether QUERY is on or off if an error is encountered, or no message if LOCK has not been specified. Other messages may be displayed with any response needed being immediately obvious.

Using the Hard Drive as the SYSTEM Drive

It is easy to use one of your hard drive partitions as a "system" drive. This means that the drive must contain all of the /SYS files and be referenced as the "zero" drive. In order to do this, the procedure takes three steps.

First you need to transfer all of the files from a floppy "system" diskette to the hard drive partition that uses head zero. Do this via the command:

```
BACKUP :s :d (S,I)
```

This will transfer all files from the floppy identified by ":s" to the logical hard drive identified by ":d".

Second, make the hard drive the "system" drive via the command:

```
SYSTEM (SYSTEM=d)
```

which will change the hard drive identified by ":d" to now be referenced as logical drive zero, ":0" and the previous drive zero will now be referenced with the logical drive number previously assigned to the hard drive, ":d".

If you want to configure the hard drive as drive zero so that the next time the system is booted, the hard drive will automatically come on line as drive zero, you can do this with the following procedure:

```
LDOS 5.3: SYSTEM (SYSGEN,DRIVE=d)
LS-DOS 6.3: SYSGEN (DRIVE=d)
```

In either case, "d" represents the drive number of your first floppy after the above swap is performed. That floppy system disk becomes your "startup" disk to be used when you BOOT your machine.

Using the SubDISK facility

Creating a SubDISK

The SDFORM utility creates a SubDISK file based on information given by you. You can create a SubDISK file from the DOS, from inside BASIC, or from within an applications program (in most cases). The syntax to create a SubDISK file is:

SDFORM [filespec (parm, parm, ...)]

filespec	This is the name of the SubDISK file to be created. Standard operating system rules regarding file names apply.
Type=t	't' is one of 1, 2, 5 or 8. This indicates the type of SubDISK to be created. The default is 5.
Sides=s	's' is either 1 or 2, indicating the number of "logical" sides to assign to the SubDISK file. The default is 1.
Density=d	where 'd' is either "D" for double density, or "S" for single. This is the "logical" density. The default is "D".
Cylinders=c	where 'c' is a value between 3 and 96. This indicates the number of "logical" cylinders to allocate within the SubDISK file. The defaults are 40 for type 5, and 77 for type 8.
Notes:	SIDES and DENSITY have no meaning in regard to types 1 and 2. CYLINDERS must be specified for types 1 and 2. All parameters may be abbreviated to their first character. Items within "[]" are optional.

Thus, a typical format command is structured as:

```
SDFORM filespec (optional parameters)
```

The "filespec" entry is the destination file specification of the SubDISK file. If "filespec" is not specified, a prompt will appear asking for it. If sufficient disk space is not available, no SubDISK file will be created, and the SubDISK utility will abort, displaying the error message:

```
Disk Space Full
```

If a file already exists by that name, the utility will abort, displaying the error message:

```
File Already Exists
```

There are four optional parameters that can be used in creating a SubDISK: Type, Sides, Density, and Cylinders.

TYPE is entered as either 1, 2, 5 or 8 (the default is 5), indicating the type of SubDISK. A type 1 SubDISK uses an optimized storage technique, allocating files in the smallest blocks as possible (1 sector per granule, 8 granules per cylinder). Using type 1, a file will only use the minimum disk space possible under the DOS. However, it is not possible to use a type 1 SubDISK as a system disk. The maximum size of a type 1 SubDISK is 192K. A type 2 SubDISK uses larger blocks, but can also be twice as large as type 1 (2 sectors per granule, 8 granules per cylinder). Types 5 and 8 refer to the type of floppy drive to simulate as a SubDISK (5" or 8").

SIDES is entered as a value of 1 or 2 (the default is 1), indicating the number of logical sides (1 or 2) to allocate in a SubDISK file.

DENSITY is entered as "D" for double density, or "S" for single density. This refers to the "logical" density within the SubDISK file. The default is double density. CYLINDERS is entered as a value between 3 and 96 indicating the number of "logical" cylinders to allocate within the SubDISK file. The default value is 40 for 5" drive types, or 77 for 8" drive types.

SIDES and DENSITY may not be used with types 1 and 2. CYLINDERS must be specified if type is 1 or 2. All parameter names may be abbreviated to their first character. If no parameters are entered when creating a SubDISK, then prompts will be displayed requesting the parameter values.

Pressing <ENTER> in response to a prompt will indicate use of the default value for that parameter. Invalid responses will cause reprompting until valid values are entered. If one or more of the parameters are entered on the command line, no prompts will appear, and the default values for parameters not entered will be used. Pressing <BREAK> to any prompt will cause the create utility to abort.

Possible error messages from SDFORM

Requested file would require more than 12 extents ! - This error message will occur if the SubDISK file to be created would be broken into an excessive number of non-contiguous pieces. This is normally indicative of a diskette that is very fragmented. Usually a BACKUP-BY-CLASS to an empty, formatted disk will reduce this fragmentation.

Parameter Error - One or more of the parameters entered when invoking SDFORM were either incorrectly specified or misspelled.

File Already Exists - The filename specified is already in use on the specified destination drive.

Setting up an existing SubDISK file

Once you have created SubDISKs, you use the SD utility to make the disk drive emulation accessible from a drive slot. SD is invoked with the syntax:

SD :drivespec filespec (Disable)

drivespec	The logical drive slot in which to install the SubDISK file. The colon is optional.
filespec	Specification of the SubDISK file to installed.
Disable	Disables an existing SubDISK file.
Abbrev:	D=Disable

The SubDISK setup utility installs a driver program into high memory (or low memory in LS-DOS 6.x, if available) which interfaces the SubDISK file with the system. In addition to the driver program, one linkage module is installed for every SubDISK drive used. A SubDISK may be re-installed using the same logical drive number as often as desired. No additional memory will be used on subsequent installations. Therefore, it is possible to set-up a SubDISK, BACKUP files to it, set-up another, BACKUP files to it, etc. It is also possible to SYSGEN with an active SubDISK, but the SubDISK file must be present on the same drive it was SYSGENed on, and in **EXACTLY** the same physical position, or disastrous results will occur.

THE "drivespec" entry is the logical drive number (1-7) to use as the SubDISK. The colon is optional. If the drivespec is not specified, or is invalid, a prompt will appear asking for it.

The "filespec" entry is the file specification of the SubDISK file. The default extension for SubDISK files is "/DSK". If "filespec" is not specified, a prompt will appear asking for the filespec. Obviously, "filespec" cannot be on the same drive number as "drivespec". If this should happen accidentally, the message:

SubDISK drivespec identical to FILE drivespec

will be displayed, and the assignment will be aborted.

Most commands available from the DOS can be used on the SubDISK drive. Do not use any form of the REMOVE, PURGE or KILL commands on a file that is serving as an active SubDISK drive. Disable the SubDISK drive prior to removing the file.

Also, do **NOT** attempt to "COPY filespec1 to filespec2", where "filespec2" is an active SubDISK file. This is not the proper method of swapping active SubDISKs. LS-DOS 6.x keeps active SubDISK files open, so the error "File Already Open" may be displayed on certain functions attempting to access the file.

Generally speaking, don't do anything which will modify an active SubDISK file, or a logical drive slot (DCT) that is currently assigned to a SubDISK.

SubDISKs can be used as system disks within certain limits:

The SubDISK file to be used as a system disk cannot be a type 1 SubDISK. Only type 2, 5, or 8 can be used as system disks. Type 2 will provide for the most efficient storage of the operating system.

The SubDISK to be used as a system disk cannot be located on the physical drive or SubDISK that is currently serving as drive 0 (the system drive). Also, no currently active (installed) SubDISKs may be located on the current drive 0 or on the destination disk or SubDISK when executing **ANY** form of the SYSTEM (SYSTEM=d) command.

Possible error messages from SD

Parameter Error - One or more of the parameters entered when invoking SD were either incorrectly specified or misspelled.

Can't install SubDISK, memory not available - An attempt was made to alter a SubDISK configuration that would require additional memory, and this additional memory is not available. This will occur if an attempt is made to install an additional SubDISK from inside of BASIC, for example. However, swapping of SubDISK files in already installed drive slots is allowed.

Drive not a SubDISK - The SD disable parameter was specified, and the indicated drive slot was not an active SubDISK.

File Contains too many extents - Any SubDISK file is limited to a maximum of twelve directory extents. To this end, SDFORM will not allow creation of a SubDISK with more. If a SubDISK file is copied from one diskette to another, it is possible that the destination file will have more than twelve extents. To avoid this, copy large SubDISK files only to relatively "empty" diskettes, and check the number of extents on the destination file with the "DIR :d (A)" command.

Unable to reclaim both driver and linkage memory - This is a non-fatal message. It simply indicates that a disable command was issued, and SD could not totally free memory. There are two major causes for this. First, there was more than one SubDISK active. The driver cannot be removed because it must service the other SubDISK. Second, some program, driver or filter used after SD is using memory. SD cannot release its memory

because it has been "trapped" by this second program. The best way to recover this memory is to re-boot.

Illegal SubDISK file - The file specified for installation with SD was not a SubDISK file created by SDFORM.

SubDISK drivespec identical to FILE drivespec - An attempt was made to install a SubDISK file into a logical drive slot occupied by the media or SubDISK containing the SubDISK to be installed.

Disabling a SubDISK

The syntax to DISABLE a SubDISK is:

```
SD :drivespec (Disable)
```

"D" may also be used as an abbreviation for DISABLE. If "drivespec" is omitted, a prompt will appear asking for it. Disabling a SubDISK drive does not affect the contents of a SubDISK file. The main purpose of disabling the SubDISK is to reclaim memory used by the driver/linkage, and close the SubDISK file. If it is not desired to reclaim the memory used by the linkage, simply use the DOS command "SYSTEM (DRIVE=d,DISABLE)". Sometimes it is not possible to reclaim the memory used by the driver. This may be the case if some module was installed after the SubDISK, or additional SubDISKs are operational.

Examples of SubDISK use

To create a SubDISK file "DRIVE5/DSK:3", with 30 cylinders and optimized allocation (Type 1):

```
SDFORM DRIVE5:3 (C=30,T=1)
```

Note that a default extension of "/DSK" will be added to the filespec given unless the filespec contains an extension, or the filespec contains a null extension.

To install the SubDISK as logical drive 5:

```
SD 5 DRIVE5:3
```

Again, note that the default extension is "/DSK". The syntax for creating a 5" double-sided, double density SubDISK file named SDFILE is:

```
SDFORM SDFILE/ (S=2)
```

To install it as logical drive 7, type:

```
SD :7 SDFILE/
```

Note that in both cases, a "/" with nothing after it was used to force a filename with no extension.

To create a mirror-image of a floppy disk on a drive of larger capacity, simply create a SubDISK that exactly matches the desired diskette in cylinders, density and number of sides. Next, install the SubDISK with SD. Now, simply perform a mirror-image backup from the floppy disk to the SubDISK.

This SubDISK can now be used to make floppy disk copies (through mirror-image backup, or another suitable utility), or for almost any other purpose.

For example, let's say that a 5", double density, 40 cylinder diskette is to be placed on a Hard Disk system, which is set up on drive 5. Drive slot 7 is available. First, create the SubDISK with:

```
SDFORM SUBDISK:5 (C=40,D="D")
```

Note that the defaults of 5" and 1 side will be used, and the extension "/DSK" will be assigned. Second, install the SubDISK with the command:

```
SD :7 SUBDISK
```

This will install the file in logical drive slot 7. Now, assuming that drive 3 is a floppy disk, the DOS command:

```
BACKUP :3 :7
```

would make a mirror-image copy of the floppy disk in drive 3. Note also that if SDFORM is entered without giving any parameters, all necessary information will be prompted for, as shown in the following examples:

```
SDFORM - SubDISK Creation Utility - Version
x.x.xx
Copyright (C) 1983/89 by MISOSYS, Inc.

Enter Filespec : SubDISK:7
Enter Disk Type <1, 2, 5, or 8> [5] : 1
Enter Cylinders (Note : 1 Cyl =  2.00K) : 50
SubDISK File Created

Enter Filespec : diskdis2:7
Enter Disk Type <1, 2, 5, or 8> [5] : 5
Enter Sides     <1,2> [1] : 2
Enter Density   <S,D> [D] :
Enter Cylinders (Note : 1 Cyl =  9.00K) : 12
SubDISK File Created
```

If only the <ENTER> key is pressed in response to a prompt, the default value for that parameter will be assumed. If there is no default value, the prompt will re-appear.

Backing up the Hard Drive - Using ARCHIVE

The ARCHIVE utility is designed to provide a controlled archival of one or more files from a large capacity storage device to a smaller floppy disk drive. It is invoked as follows.

ARCHIVE source :d (Query=N,Size=nn,Test,Verify)

source	Specifies the filespec or drivespec of the file or drive you wish to archive. If a filespec is entered, only that file will be archived. If only a drivespec is entered, all files of length greater than the destination floppy size will be archived (see text on the SIZE parm).
d	Specifies the drive which will receive the archived segments.
Query=N	If a "source" drivespec was entered, then you will be prompted with each source filename to archive. If you do not want the prompts, then specify Q=N.
Size=nn	If a "source" drivespec was entered, then only files that are larger than the capacity of the destination drive will be selected for archival. If you wish to alter this threshold, enter a new size. The value is entered in quantities of sectors. Entering <Size=0> will select all files for archival.

continued

	continuation
Test	Since any given archival requires a set of formatted floppy diskettes on hand, you may wish to know how many floppies are required for an archival set. If you specify this TEST parameter, ARCHIVE will report on the quantity of diskettes needed.
Verify	Normally, ARCHIVE does not verify sectors written to the floppy diskettes. If you wish to force a verify operation, specify "Verify" when you invoke ARCHIVE.
Abbrev:	Q=Query, S=Size, T=Test, V=Verify

The hard drive partition is backed up to floppies. You may use the BACKUP utility provided with your DOS to copy the hard disk files onto floppy diskettes. However, when a single hard drive file exceeds the total size of a floppy diskette, the ARCHIVE utility can be used to segment the file and transfer each individual segment to a corresponding floppy diskette. The RESTORE utility is then used to restore each segment when you need to reclaim the backup copy.

All segments of all files that are archived include a header record generated by the ARCHIVE utility. This header record is used essentially by the RESTORE utility to enable it to regenerate an exact image of the original file from its segments. The header record is also used to identify each diskette included in an archive set. The contents of the header record are detailed in the TECHNICAL INFORMATION SECTION. If there is a problem in reading one or more sectors of a source file, information on those sectors is placed into the appropriate header record and may be needed by you in a data recovery operation. Since each file segment has this header record introduced, the archived file segments cannot be directly used to BACKUP to the source file via the DOS's BACKUP utility. The RESTORE utility provided with this package is the tool to restore a hard drive from its archived files.

ARCHIVE will only select files that are classified as visible. Also, the visible files on the hard disk must be greater in size than the floppy used for the archive process in order to be selected. Since ARCHIVE only selects files of size greater than the destination floppy, where your hard drive intermixes both small and large files, you may need to use both the DOS's BACKUP and ARCHIVE to make archival copies of all your hard disk drive files. Since using two utilities may prove cumbersome, you may want to use the SIZE parameter with a value of zero to force ARCHIVE to select all visible files. Bear in mind that the archived files must be RESTOREd prior to their use.

There is no requirement that you must restore an archived file to the drive that the source was on. Archived file sets may be restored to other drives providing that the size of the drive is sufficient to handle the file or files. This provides a method to transfer large files from one hard disk drive to another.

During the archival process, ARCHIVE will display various messages - some of which require a response from you. The meaning of each message and any response required will now be discussed.

```
Enter filespec or drive to archive? >
.........

Source filespec requires DRIVESPEC...
```

These messages indicates that you either omitted the "source" entry on the command line or entered an erroneous filespec. If a filespec is entered, it must include the drive specification. No password is required and none should be entered. For a single-file archival, enter its file specification. If you wish to select the files to archive by query, then enter a drive specification (in the form, <:s> where "s" is the logical drive number).

```
Enter destination drive <1-7>? > .
```

This message indicates that you either omitted the destination entry on the command line or entered an erroneous drivespec. Note that you cannot archive files to drive 0.

```
Archive file: FILENAME/EXT:D DD-MON-YY     ? .
```

This is the prompt that is displayed for each file when you are archiving a drive. There are three choices for a response. Choose <Y> or <y> and that file will be archived. Choose <C> or <c> and that file will also be archived. In addition, the QUERY operation will henceforth be turned off during the archival operation. Choose <N> or <n> and that file will be ignored. The DD-MON-YY display is the directory date of the file.

Destination drive is write protected - replace <ENTER>

If ARCHIVE senses one of your destination floppy diskettes as being write protected, this message will be displayed. Remove the write protection from the diskette and depress the <ENTER> key to continue. If you had software write protected the drive (via the DOS's SYSTEM command), you must abort the archival by depressing the <BREAK> key, remove the software write protection, then re-invoke the archival command.

Destination drive not ready - prepare <ENTER>

The destination drive is not available for writing. Either the diskette is not in place or the drive door is open. Prepare the drive with a floppy diskette and then depress the <ENTER> key.

Destination disk already used...

This message indicates that the diskette in place in the destination drive already has a file containing one of the segments of the file currently being archived. Change to an unused destination diskette.

This destination disk has no space available...

This indicates that the diskette in place in the destination drive has no more space for archiving a file segment. Change to an unused diskette or one with unused space available. Depress <ENTER> to resume.

Enter new destination disk and <ENTER>

This message will be displayed during the archival process when each of the destination diskettes has been completed. Change to a new disk and depress the <ENTER> key to resume the archival process with the next file segment.

Sectors to archive: xxxxx
- Archiving sector: yyyyy

This message is displayed while a file is read from the source and written to the destination. The "xxxxx" field is a count of the total number of sectors that need to be read from the source file. The "yyyyy" field is the number of the sector (relative from 1) that is currently being transferred.

Source disk read error - Sector zzzzz

This unfortunate error will be displayed if there is any error in reading a sector from the source file. The field "zzzzz" contains the relative sector number (relative from 0) of the file. The sector number will be listed in the BADREC field of the header record contained in the file segment currently being written. All bytes in the sector will be stored as an ASCII space character (decimal 32).

Cannot run ARCHIVE from JCL!

Because ARCHIVE generally requires the controlled exchange of diskettes and responses to queries, it is not designed for operation in a batch environment. If ARCHIVE senses its invocation from Job Control Language, it will display this message and abort.

Parameter error!

This message indicates that one or more parameters entered on the command line are incorrect. Re-enter the command line using correct parameters.

Archival disk count: ccccc

This message will be displayed when you have selected the TEST parameter. The disk count will be in field ccccc. The count assumes that all diskettes are identical to the one in the destination disk drive, all have totally unused space except for the DIR/SYS and BOOT/SYS files, and that ARCHIVE detects no flawed media during the subsequent archival process.

Restoring the Hard Drive - Using RESTORE

The RESTORE utility is used to reconstruct files that have been archived via the ARCHIVE utility. RESTORE combines all of the segments of an archived file into a single file. The restoral process does not require the restored file to be existing on the drive that is restored. All attributes of the original file will be restored. RESTORE can be used to reconstruct a single original file from the archived file set or it can be used to reconstruct all archived files contained in the archived file set. RESTORE is invoked via:

RESTORE source :d (Query=N,Verify)

source	Specifies the filespec or drivespec of the file or drive you wish to restore. If a filespec is entered, only that file will be restored. If only a drivespec is entered, all files in the archived file set will be selected for restoral.
d	Specifies the destination drive which will receive the reconstructed file(s).
Query	If a "source" drivespec was entered, then you will be prompted with each source filename to restore. If you do not want the prompts, then specify Q=N.
Verify	Normally, RESTORE does not verify sectors written to the hard drive. If you wish to force a verify operation, specify "VERIFY" when you invoke RESTORE.
Abbrev:	Q=Query, V=Verify

An archived file set may contain one or more diskettes - each containing a segment or segments of archived files. If you want to reconstruct all of the files archived in the file set, specify the drivespec of the drive that will be used to read the diskettes. If you only want to reconstruct one of the files

contained in the archived file set, enter its complete filespec as the "source" entry.

RESTORE posts messages concerning its progress in the reconstruction process and prompts for diskette exchanges when it completes the reading of all file segments contained on a diskette that were selected for restoral. RESTORE keeps track of which diskettes have been completed and will automatically terminate the restoral process once it recognizes that the reconstruction is complete. The meaning of the messages displayed and the appropriate responses required for entry by you are now discussed.

```
Enter SOURCE filespec or drive? >
.............

SOURCE filespec requires DRIVESPEC...
```

One of These messages will be displayed if you omit the filespec of the file you wish to restore or drivespec of the disk drive to be used to read the archived file set. Respond by entering the requested information.

```
Enter DESTINATION drive to restore <0-7>? > .
```

The display of this message means that you have either omitted the specification of the drive to write the reconstructed file(s) or have entered an erroneous drive number. Respond by entering the requested information.

```
Restore file: FILENAME/EXT:D DD-MON-YY     ? .
```

This is the prompt that is displayed for each file when you are restoring a drive. There are three choices for a response. Choose <Y> or <y> and that file will be restored. Choose <C> or <c> and that file will also be restored. In addition, the QUERY operation will henceforth be turned off during the restoral operation. Choose <N> or <n> and that file will be ignored on diskettes of the file set that it appears.

```
DESTINATION drive is write protected -
replace <ENTER>
```

If RESTORE senses your destination hard disk drive as being write protected, this message will be displayed. Remove the write protection from the drive and depress the <ENTER> key to continue. If you had software write protected the drive (via the DOS's SYSTEM command), you must

abort the archival by depressing the <BREAK> key, remove the software write protection, then re-invoke the restoral command.

SOURCE drive not ready - prepare <ENTER>

The source drive is not available for reading. Either the diskette is not in place or the drive door is open. Prepare the drive with a floppy and then depress the <ENTER> key.

DESTINATION drive not ready - prepare <ENTER>

The destination drive is not available for writing. Either you referenced a non-existent drive or you referenced a removable cartridge drive that is not active. Prepare the drive and then depress the <ENTER> key or abort by depressing the <BREAK> key.

SOURCE disk already used...

This message indicates that the diskette in place in the source drive has already been restored. Change to an unused source diskette.

SOURCE disk from different archive set...

This indicates that the diskette in place in the source drive has a file segment with the same name as one currently being reconstructed; however, it does not belong to the same archived file set. Change to a proper diskette in the archived file set currently being restored. Depress <ENTER> to resume.

Enter new SOURCE disk and <ENTER>

This message will be displayed during the restoral process when each of the source diskettes has had all of its selected file segments completed. Change to the next disk and depress the <ENTER> key to resume the restoral process.

Sectors to restore: xxxxx
- Restoring sector: yyyyy

This message is displayed while a file is read from the source and written to the destination. The "xxxxx" field is a count of the total number of sectors that need to be read to complete the reconstruction of the original file. The

"yyyyy" field is the number of the sector (relative from 1) that is currently being transferred.

Source disk read error - Sector: zzzzz

This unfortunate error will be displayed if there is any error in reading a sector from the source file. The field "zzzzz" contains the relative sector number (relative from 0) of the file. All bytes in the sector will be restored as an ASCII space character (decimal 32).

DESTINATION disk write error: hh
- Sector: zzzzz

This error message will be displayed if RESTORE detects an error when writing a sector to the destination disk. Field "hh" contains the DOS error code in hexadecimal while field "zzzzz" contains the number of the sector (relative to 0) that was written.

Out of memory - can't continue!

Each file that is being reconstructed requires a 128-byte buffer maintained in memory. As a file is completely restored, it frees up this buffer. Thus 128xN bytes of memory are required where "N" is the number of files being actively reconstructed. It would be most unusual to obtain this error message. It would most likely mean that a large part of your machine's memory is unavailable - perhaps being used as a spool buffer or other abnormally extra usage. Attempt to free up the high-memory space and attempt the RESTORE operation again.

Cannot run RESTORE from JCL!

Because RESTORE generally requires the controlled exchange of diskettes and responses to queries, it is not designed for operation in a batch environment. If RESTORE senses its invocation from Job Control Language, it will display this message and abort.

Parameter error!

This message indicates that one or more parameters entered on the command line are incorrect. Re-enter the command line using correct parameters.

Operating the hardware clock

Date and time operation

One of the options available with your hard drive is the hardware clock interface. This option utilizes a state of the art clock module which provides calendar date and time keeping. The clock module also incorporates an alarm function which can be programmed to generate interrupts to the operating system at a specified time. This alarm checking is performed with absolutely no use of your computer's processor resources. The clock module retains its date and time values by means of an imbedded lithium battery; battery life is approximately ten years.

The hardware clock also includes 50 bytes of user accessible RAM. This RAM is port-accessed and non-volatile, which means that anything stored in that RAM is retained after the power is turned off.

The primary utility program used to access the clock is MCLOCK/CMD. It is invoked as follows:

MCLOCK (Set,Dst,CLOCK)

Set	Specifies that you want to set the date or time stored in the clock. Set defaults to "NO"
Dst=Y|N	This specifies that you want the clock to automatically adjust to daylight saving time (see text). Dst has no default. Dst must be used in connection with SET.
CLOCK=N	If the clock module is going to be unused for a long period of time (say years), the usable life of the lithium battery can be lengthened by turning the clock off. This will, of course, erase any data stored in the clock. The default is "ON".
Abbrev:	Set=S,Dst=D

Anytime MCLOCK is invoked, it will read the current values of date and time from the clock and update the system date and time values maintained by the DOS. Thus, the primary means of obtaining the date and time is to invoke MCLOCK as either an AUTO command at BOOT or as part of a startup job control language job stream. Also, to suppress the normal "DATE?" and "TIME?" prompts generated by the operating system, it will be necessary to invoke the following two commands once you are using your BOOT disk as the "system" disk:

```
SYSTEM (DATE=OFF)
SYSTEM (TIME=OFF)
```

This will then inhibit the DOS from asking you to manually enter the date and time; it will be read from the clock upon invoking MCLOCK. Once the values are obtained from MCLOCK and the values stored in the system, the DOS, using your machine's "realtime clock" will maintain the time.

There are other methods of reading the data values from the clock. Any other method than the one described herein requires patching your operating system. Any optional patches, if available, are described in the README/TXT file on the software distribution disk.

If your system does a poor job of maintaining the time accurately, it may be necessary to periodically update the system time from the clock by subsequent invocations of MCLOCK, or by patches to the DOS reprogramming it to periodically re-read the clock values. Consult any additional text on this subject in the README/TXT file.

As supplied by MISOSYS, the MCLOCK utility program sets a bit to "one" in the clock module which establishes the daylight savings time option. The Daylight Savings Enable (DSE) bit is a read/write bit which enables two special updates when DSE is set to one. On the last Sunday in April the time increments from 1:59:59 AM to 3:00:00 AM. On the last Sunday in October when the time first reaches 1:59:59 AM it changes to 1:00:00 AM. These special updates do not occur when the DSE bit is a zero. This bit is not affected by internal functions or RESET. Thus, once the option is established, it is retained. Note that any resetting of the clock time will use the current value of the "DSE" bit stored in the clock. If you want to change the current setting, you must enter either "D=ON" or "D=OFF", along with the "Set" parameter.

The clock module contains 64 bytes of memory, the first fourteen of which are used as registers and time storage locations. Of these fourteen, one is used to generate a known value as long as the internal lithium battery has sufficient power to support the clock. When MCLOCK is invoked, it first reads that value. If the value is invalid for a properly powered clock, the error message,

Clock is either not installed or battery is exhausted!

is displayed. Battery life is approximately ten years. If the message seems premature, it may be that your hard drive is not plugged into the 50-pin expansion port of your computer. If the battery is truly exhausted, then the clock module needs to be replaced. The module is a Dallas Semiconductor part DS1287. Contact MISOSYS for a replacement.

Clock memory addressing

Since only fourteen of the 64 available bytes of memory are used by the clock, the remaining fifty are for your use - for whatever purpose you can define. A clock memory location is accessed through two machine ports: one to specify the memory address (which of the 64 bytes entered as a number from zero to sixty-three) and another to actually transfer the data. The following assembly language code illustrates access of one memory byte:

```
DI                      ;Interrupts off
LD    A,14
OUT   (176),A           ;Specify address 14
IN    A,(177)           ;Read the value
EI
```

This example read the value stored at address 14. Note that the machine interrupts must be turned off so that no operation occurs between the time the address is passed to the clock and the data value is read or written. An appropriate code stream to write the value 27 to the last memory address in the clock would be:

```
DI                      ;Interrupts off
LD    A,63
OUT   (176),A           ;Specify address 63
LD    A,27
```

```
OUT   (177),A          ;Write the value
EI
```

This same result can be accomplished from BASIC using INP and OUT instructions. Since BASIC provides no direct way to disable the machine's interrupts, one must use a "brute force" method as follows:

```
100 IA = &H38: REM Model III use "&H4012"
120 POKE IA,&HC9
140 OUT 176,14:BB=INP(177)
160 POKE IA,&HC3
```

This duplicates the first assembly language example.

Operating the clock alarm function

The clock module has three distinct functions of generating interrupts to the machine's processor. One function generates an interrupt every time the clock updates its internal registers to the next second (UIE). Another is a periodic interrupt adjustable under software control from 500 milliseconds to 122 microseconds (PIE). And the third function is to generate an interrupt based on a programmable time entry (AIE). MISOSYS has provided a utility program to operate these three interrupt functions; however, perhaps only the latter alarm function will be practical for use unless you want to interface your own programs to the alarm interrupts.

MALARM can be used to set the alarm time, enable or disable the alarm, set and enable the periodic timer, and enable or disable the update interrupt. This is done by invoking MALARM as follows:

MALARM (Set,Disable,Enable,Pie,Uie,OFF)

Set	Specifies that you want to set the alarm time stored in the clock. Set defaults to "NO"
Disable	This specifies that you want to turn off the alarm interrupt.
Enable	This specifies that you want to turn on the alarm interrupt.
Pie=0...15	This specifies that you want to enable or disable the program mable timer. A value of zero turns it off; values of one through 15 establish the interrupt rate according to the "PIE table".
Enable	This specifies that you want to turn on the alarm interrupt.
Uie=Y|N	Specifies that you want to enable (Y) or disable (N) the update interrupt which would be generated every second.
OFF	This specifies that you want to disable all three interrupts and remove the task routine from memory.
Abbrev:	Set=S,Disable=D,Enable=E, Pie=P,Uie=U

Although the clock can be programmed to generate interrupts under preset conditions, some other code must make use of those interrupts and provide something tangible (i.e. visibal or audable). Without trying to get sophisticated, MALARM installs a small 47-byte task processing routine to service these interrupts. The task routine will write an "A" character to the video screen at the last column of the first row, alternating the "A" with a blank on each interrupt. Thus it requires a continuous interrupt to observe a blinking character.

When you set an alarm time and enable the alarm, an interrupt will be generated whenever the hours, minutes, and seconds of the clock time match the hours, minutes, and seconds of the alarm time. This would occur once each day; there is no date alarm entry. However, the hours-minutes-seconds entry fields also can be set with a "don't care" or "match all" code. This means that a field, or fields, can be set with the "don't care" value and the alarm will interrupt whenever the other fields match. As an example, setting the alarm time to "xx:15:00" will cause an interrupt at fifteen minutes past every hour. Since the alarm task provided only changes its visual state on each interrupt, you would probably not notice such an alarm. Therefore, an alarm time value of "xx:15:xx" would generate 60 interrupts - one for each second while the clock time was running from xx:15:00 through xx:15:59. That you may notice!

Whenever MALARM is invoked, it will display the current alarm setting. When you specify the "SET" parameter, MALARM will prompt for your entry with the query,

```
Enter new time [HH:MM:SS] or <ENTER> to
continue:
```

When you make your entry, a "don't care" code is designated by entering a "99" instead of the normal values for hours (00-23) minutes (00-59) and seconds (00-59). "Don't care" codes are displayed as "XX" whenever the current alarm setting is displayed.

The periodic interrupt is disabled with a value of 0; any value from 1-15 generates interrupts according to the following table of values:

PIE	Interrupt Rate
0	None
1	3.90625 ms
2	7.8125 ms
3	122.070 us
4	244.141 us
5	488.281 us
6	976.5625 us
7	1.953125 ms

PIE	Interrupt Rate
8	3.90625 ms
9	7.8125 ms
10	15.625 ms
11	31.25 ms
12	62.5 ms
13	125. ms
14	250. ms
15	500. ms

Technical note for programmers only: The task also has a means for programmers to tie into the task if they care to write additional processing routines. The task installed into memory has the following header protocol:

```
TSKBGN    JR   TSKDO
     DW   $-$
     DB   5,'ALARM'
TASKX     RET
     DW   0          ;Possible user vector
TSKDO...
```

The task exits by means of the RET instruction coded at TASKX. To tie into the task, place your entry vector into the two bytes immediately following TASKX and change the TASKX code to a JP instruction (X'C3'). When control is passed to TASKX, register B is loaded with the clock's flag register C. Bit 6, 5, and 4 will be a one corresponding to an interrupt from PIE, AIE, or UIE respectfully.

Alternatively, the programmer could take over the entire external I/O interrupt vector which would be used solely by the clock module. For additional details concerning the clock's interrupts, consult a Dallas Semiconductor manual on the DS1287 chip.

As a footnote, the UIE interrupt is useful to limit reading the clock values to only when the value changes each second.

Joystick operation

The joystick option adds a port to read the joystick, and includes an inexpensive joystick. The joystick port uses a standard DB9 connector. The joystick can be altered to operate in either four or eight positions by means of a lever switch located on the bottom of the joystick.

The optional joystick interface should look to computer programs just like the old "Alpha Products" joystick addressable from port 0. In this configuration, the joystick switch positions and fire button are mapped to bit positions as follows:

bit	operation
0	up
1	down
2	left
3	right
4	fire

Note that the stick operation produces inverted logic; normally, a position is turned ON if the corresponding stick operation is not active. The following BASIC program illustrates one method of reading the joystick port.

```
10 CLS:PRINT CHR$(15);
15 X=0:Y=0:OLDX=0:OLDY=0
20 PRINT@(1,17),"Test of joystick port on
SCSI H/A"
30 PORT = 0+256:REM add 256 to avoid conflict
with XLR8er
40 BYTE = INP(PORT) XOR 255
50 GOSUB 1000
60 PRINT@(3,20),"Joystick value is: "+BYTE$
65 MOVE$=""
70 IF (BYTE AND 2) THEN MOVE$ = MOVE$+
 "UP        ":Y=Y-1
80 IF (BYTE AND 1) THEN MOVE$ =MOVE$+
 "DOWN      ":Y=Y+1
90 IF (BYTE AND 8) THEN MOVE$ =MOVE$+
 "LEFT      ":X=X-1
100 IF (BYTE AND 4) THEN MOVE$ =MOVE$+
```

```
 "RIGHT     ":X=X+1
110 IF (BYTE AND 16) THEN MOVE$ =MOVE$+
 "FIRE      "
120 IF MOVE$="" THEN MOVE$ = "No Action"
150 PRINT@(5,20),"Stick action is: "+
MOVE$,," "
152 IF X<0 THEN X=X+80
153 IF X>79 THEN X=X-80
154 IF Y<0 THEN Y=Y+24
155 IF Y>23 THEN Y=Y-24
156 IF (X<>OLDX) OR (Y<>OLDY) THEN GOSUB 1100
160 GOTO 40
1000 BYTE$="00000000": REM initialize bit
pattern to all zeroes
1010 FOR I = 0 TO 7
1020 IF (BYTE AND 2^I) THEN MID$(BYTE$,8-
I,1)="1"
1030 NEXT
1040 RETURN
1100 PRINT@(OLDY,OLDX)," ";
1110 PRINT@(Y,X),"*";
1120 OLDX=X:OLDY=Y
1130 RETURN
```

Using the joystick filter

Obviously, without some form of program modification, existing programs cannot make use of the joystick. However, for that very reason, MISOSYS provides a keyboard filter which reads the joystick and converts stick operations to standard keystroke values as if they were typed from the keyboard. The filter enables you to automatically generate five different values from the joystick. Use of the filter may make the joystick practical for use with programs which utilize the ARROW keys for cursor manipulation, and which use the DOS keyboard driver. The filter is invoked via the following syntax:

LS-DOS: **SET *JS JSTICK (Up=dd,Down=dd, Left=dd,Right=dd,Fire=dd,repeaT=d)**
FILTER *KI *JS

LDOS: **FILTER *KI JSTICK (Up=dd,Down=dd, Left=dd,Right=dd,Fire=dd,repeaT=d)**

Up=dd	The value generated by the UP stick position; defaults to UP ARROW.
Down=dd	The value generated by the DOWN stick position; defaults to DOWN ARROW.
Left=dd	The value generated by the LEFT stick position; defaults to LEFT ARROW.
Right=dd	The value generated by the RIGHT stick position; defaults to RIGHT ARROW.
Fire=dd	The value generated by the FIRE BUTTON; defaults to ENTER.
repeaT=d	Establishes a rate for automatic repeating of the "keystroke" as long as the switch position is kept activated; the rate defaults to 3 and can be entered as any number from 0 (fastest) to 9 (slowest).
Note:	dd is a decimal value; enter in hexadecimal as "X'xx'".
Abbrev:	U=Up,D=Down,L=Left,R=Right, F=Fire,T=repeaT.

The repeat rate can also be changed from the joystick after the filter is installed and activated. If moved to the 8-position operation, simultaneously operating the joystick to the UP and LEFT positions and then depressing the FIRE BUTTON will increase the repeat rate. Simultaneously operating the joystick to the DOWN and RIGHT positions and then depressing the FIRE BUTTON will decrease the repeat rate.

Altering character values

Once the joystick filter is installed, any of the character values generated by stick positions, as well as the repeat rate may be altered by using the SETJS utility. This is invoked as follows:

SETJS (Up=dd,Down=dd,Left=dd,Right=dd, Fire=dd,repeaT=d)

Up=dd	The value generated by the UP stick position.
Down=dd	The value generated by the DOWN stick position.
Left=dd	The value generated by the LEFT stick position.
Right=dd	The value generated by the RIGHT stick position.
Fire=dd	The value generated by the FIRE BUTTON.
repeaT=d	Establishes a rate for automatic repeating of the "keystroke" as long as the switch position is kept activated; can be entered as any number from 0 (fastest) to 9 (slowest).
Note:	dd is a decimal value; enter in hexadecimal as "X'xx'".
Abbrev:	U=Up,D=Down,L=Left,R=Right, F=Fire,T=repeaT.

Any parameter which is not entered is left unchanged from its previous setting. The SETJS utility will display both the current and revised character values. If the program is invoked with no parameter entry, only the current values are displayed.

Technical Information

Drive Control Table Usage

This section covers the use made of fields in the Drive Control Table record assigned to a hard drive partition.

DCT+3, Bit-3

This bit is set to indicate the logical drive is a hard drive.

DCT+3, Bit-2

This bit is RESET if the drive is a REMOVABLE cartridge; otherwise, it is set.

DCT+3, Bits 1-0 and DCT+5

These fields contain the number of the starting cylinder for the logical partition. If DCT+4, bit-5 is SET, then this number represents half the actual value.

DCT+4, Bit-5

This bit, termed the DBLBIT, will be SET if you requested more than 203 cylinders for a single logical drive. In this case, a logical cylinder is composed of two physical cylinders.

DCT+4, Bit-4

This bit is set to indicate that the controller does not supply an index pulse indication to the disk driver on each rotation of the drive surface.

DCT+4, Bits 3-1

This field contains the starting head number counting from zero.

DCT+4, Bit-0

This bit-field contains the drive address.

DCT+6

This field contains the highest numbered logical cylinder on the partition. This number is relative - it references the actual number of cylinders assigned to the partition and is counted from zero. If the DBLBIT bit is set, the number of physical cylinders assigned is twice this number.

DCT+7, Bits 7-5

This field contains the number of heads assigned to the partition.

DCT+7, Bits 4-0

This field contains the number of sectors per track counted from zero. The entry is hard coded with a X'1F' which indicates 32 sectors per track.

DCT+8, Bits 7-5

This field contains the number of granules allocated per physical cylinder. If the DBLBIT bit is set, the actual quantity of granules per logical cylinder will be twice this amount. The initialization program will attempt to allocate all eight granules per logical drive. It does this by dividing the total number of sectors per logical cylinder by eight with the resulting value used as the number of sectors per granule.

DCT+8, Bits 4-0

This field contains the number of sectors per granule. The initialization program will always attempt to minimize this value (see above). For example: 32 sector per cylinder (SPC) drives use 4-sector granules (1K), 64 SPC drives use 2K granules, 96 SPC use 3K, 128 SPC use 4K, 256 SPC use 8K granules.

DCT+9

This field contains the logical cylinder number of the directory. If the DBLBIT bit is set, then this number is actually half the physical number.

SubDISK file structure

The SubDISK file contains BOOT/SYS and DIR/SYS, just as any floppy does. It also contains an additional sector not accessible on a logical drive basis. Relative record X'0000' of the SubDISK file contains an ID field to distinguish it from a non-SubDISK file. It also contains DCT slot information which will be used to re-install this SubDISK next time.

SubDISK driver and linkage

The SD installation utility will attempt to install a driver program ($SD), and a linkage module ($DL), in high memory (or the low memory driver zone for LS-DOS 6.x, if room is available). The $SD driver will use the DCT for all cylinder and sector calculation. Therefore, it will only be installed once - even if there are multiple SubDISKs. However one $DL module will be installed for every SubDISK used. The driver removal program will reclaim memory IF POSSIBLE. This is a BIG if! If there is ONLY one SubDISK active, and nothing was installed underneath it (if installed in high memory), or above it (if installed in the driver area on LS-DOS 6.x), then the memory will be reclaimed. If there is more than one SubDISK active, the driver WILL NOT be disabled, because it is necessary to service the other SubDISKs.

Archive file set header record

Each file segment of an archived file has a header record written as the first sector of the file segment. The information contained in the header is used to support fail-safe reconstruction of the original file. This section covers the information contained in the fields of the header record. Each like-named file in the file set will have a header record where the first forty bytes are identical. The numbers contained within angle brackets are sector offset values in hexadecimal.

DIRECTORY <00 - 15>

This 22-byte field contains the first twenty two bytes of the original file's primary directory entry record. Any technical manual on the operation of the DOS will detail these bytes.

DATE <16 - 1D>

This field contains the date that the file set was created. The date is in standard ASCII string notation: MM/DD/YY.

TIME <1E - 25>

This field contains the time that the file set was created. The time is in standard ASCII string notation: HH:MM:SS.

RANDOM <26 - 27>

This field contains a 15-bit random number.

DISKNUM <28 - 29>

This field contains the 16-bit number of the disk set for a given file segment. Each file that is archived begins its header with a disk number of zero. The value is stored in standard low-high order.

RECBGN <2A - 2B>

This field stores the relative record number (relative to zero) of the first record contained in this file segment.

RECEND <2C - 2D>

This field stores the relative record number (relative to zero) of the last record contained in this file segment.

SPARE <2E - 3F>

This 18-byte region is currently unused. It is always filled with zeroes.

LOCKED <40 - 7F>

This field is composed of 2-byte subfields. Each subfield contains a file segment record number that was detected as bad on the destination disk during the archival process and was thus locked from use. A value of zero indicates no record lock (the zeroth record is the header). These numbers are used during restoral to indicate if any source sectors should be skipped.

BADREC <80 - FF>

This field is composed of 2-byte subfields. Each subfield contains a record number of the SOURCE file that was detected as unreadable during the archive process and that was written to this file segment as all spaces. This number may be useful to you if you neglected to write it down during the archival process.

Host Adaptor port assignments

Port 00H (000D) - Joystick

bit 0: UP position
bit 1: DOWN position
bit 2: LEFT position
bit 3: RIGHT position
bit 4: FIRE button
bit 5: S1, switch 1 - reserved
bit 6: S1, switch 2 - reserved
bit 7: S1, switch 3 - reserved

Port 0B0H (176D) - Clock address

Port 0B1H (177D) - Clock data

Port 0C0H (192D) - HDC data I/O

Port 0C1H (193D) - HDC data I/O with automatic ACK

Port 0C2H (194D) - HDC Status read

bit 0: REQ
bit 1: MSG
bit 2: BSY
bit 3: I/O
bit 4: C/D
bit 5: INUSE
bit 6: S1, switch 4
bit 7: JP1 - reserved

Port 0C2H (194D) - HDC Control write

bit 0: ACK
bit 1: SEL
bit 2: RST
bit 3: n/a
bit 4: n/a
bit 5: n/a
bit 6: n/a
bit 7: n/a

Host Adaptor interface

P1 - joystick

P2 - Host interface

P3 - Power connector

P4 - To Hard Disk Controller

========== MSCSI ==========
09/15/89

Due to cockpit trouble, the SD/CMD and SDFORM/CMD files placed on the 09/13/89 MSCSI6 disk was not the most recent version. This disk contains the "correct" versions of those two programs. Also, the MCLOCK/CMD program has been corrected with the change from 2AH to 36H at 27A8H; the SYS3CLK6/FIX was also touched.

09/13/89 - From Roy Soltoff

Whew! What a project! You have in your hands now the culmination of many many hours of work. The sheer magnitude of this hard drive project was almost more than should be worked on by one person. Nevertheless, in spite of all the obstacles I confronted during the past year concerning this project, it is shipping!

The volume of software provided herein is certainly more than was expected. A set of source code listings for EACH machine implementation is 1.5" thick of 20lb paper. Luckily, I generated the listings on a laser printer. Certainly, one should expect a few bugs to crop up in the software. If you experience a problem, please document it fully to me. I will do my best to resolve the difficulty. Pay attention to the Software manual.

As this is written the same day the first drives ship, there may be additional "README" text in the future. I'll try to keep folks abreast of vital changes.

Software notes

All files provided on the disk specific to the Model 4 operation have a filename ending with a "6". This is to guard against getting your files mixed up with those of the Model III software interface, assuming you have obtained that package.

A note about hard drives

One thing I neglected to put into the hardware manual is a word about abuse of your equipment. Please don't jar your drive unnecessarily. Take care of your equipment and it will take care of you.

Concerning the ST251-1 40 megabyte Seagate drive

Note that if you have been provided with a 40 megabyte drive, it is a Seagate ST251-1. This drive automatically parks itself when powered down; thus, there is no need to use the MSCPARK utility.

Note that this drive contains 819 usable cylinders with the Xebec controller normally used in the MHD-T34 drive kit. Four maximum-sized partitions assign 203 cylinders each. This uses a total of 812 cylinders; the last 7 cylinders are not used. If you really want to, you can bring up a fifth drive of about 320K (I don't see the purpose). If you are spliiting the drive between LDOS and LS-DOS, then the JCL assigns 105 cylinders to each system partition. This then uses up the entire drive.

Automatic installation of software

As noted in the software manual, two JCL files are provided on the disk. One is "HD20INIT/JCL" used for 20 megabyte drives, the other is "HD40INIT/JCL" for 40 megabyte drives. Each procedure will do the following:

1 Install the driver and partition the drive into 4 segments
2 Optionally add the high-level formatting to each partition
3 Backup the DOS to the "system" partition of the drive if the format is requested

Parameters available for the JCL are:

Specify FORMAT to add high-level system information
Specify DUAL for Model 4 & 3 installation
Specify NOVERIFY if you don't want MSCSIF/CMD to verify

The "DUAL" parameter is useful if you are going to utilize both Model III and Model 4 mode with your drive. By specifying "DUAL", the driver installation program refrains from using a portion of the drive used as the "system" partition under the other DOS. This, obviously requires both the Model III and Model 4 software interface.

If you want to partition your drive manually, please read the manual; attempt to understand the provided JCL; then contact MISOSYS if you are unsure how to proceed.

Note: If you just invoke the JCL with the command:

DO HD20INIT or DO HD40INIT

you will not format over any existing data stored on the drive.

If you specify "DUAL" in a JCL installation, you MUST specify DUAL everytime using another DOS unless you want to reformat your drive.

If you specify DUAL and FORMAT, only the partition specific to the Model III "system" partition will have the high-level format applied; the other three partitions will just be assigned to drive slots.

Hardware Clock updates

As referenced in the text, additional means of reading the clock date and time values have been developed. If you choose not to invoke MCLOCK at boot time, in order to read the clock, you can apply the SYSOCLK6 fix to SYS0/SYS. This will bypass the normal DOS requests for DATE? and TIME?, and obtain the appropriate values from the clock. Note that you should have the clock SET to an accurate date and time; thus, use MCLOCK first to set the data into the clock. MISOSYS makes an attempt to ship all drives that include the clock option with clock data set to ET time

(that's Eastern Time).

Even if you choose not to apply the SYSOCLK6 fix to SYSO/SYS, you may want to utilize the SYS3CLK6 fix. As the functioning of MCLOCK is to rely on the system time keeping once the actual time (and date) is read from the clock, your particular system may lose time throughout the day. This loss has to do with the frequency and duration of the machine's interrupt disabling. A patch to SYS3/SYS is included which, if applied, will obtain the CLOCK's time every instance of closing a disk file. The patch adds a short routine to the DOS's @CLOSE service call to read the time from the clock and reset SYSTEM time. This patch adds no degradation to your system, and takes up no resident memory whatsoever.

I believe that the combination of the two patches should suffice for everyone's needs. The only instance of an inaccurate dating would be if the @CLOSE service call were invoked after midnight when the system time had degraded to be before midnight. Under that scenario, the actual time would be updated but the date would lag by one day. That case would be rare. Certainly, the SYS3CLK6 fix could be revised to also read the date in addition to the time to always keep your system data tracking with the hardware clock.

========== MSCSI ==========

```
. HD40INIT/JCL Version 1.0.0 - 09/22/89
. for Model 4 MSCSI installation of 40meg drive
//if noverify
//assign parm="nostop,verify=n"
//else
//assign parm="nostop"
//end
.
. Use: DO HD40INIT (FORMAT,DUAL,NOVERIFY)
. Specify FORMAT to add high-level system information
. Specify DUAL for Model 4 & 3 installation
. Specify NOVERIFY if you don't want MSCSIF/CMD to verify
.
//PAUSE Press <ENTER> to begin the installation
. Bring up the system drive partition of hard drive
system (drive=4,disable,driver="mscsi6")
1
n
6
//if dual
105
system (drive=5,disable,driver="mscsi6")
1
y
1
106-210
y
2
106-210
y
3
106-210
y
4
106-210
y
5
106-210
y
6
106-210
//else
203
system (drive=5,disable,driver="mscsi6")
1
//end
n
6
203
system (drive=2,disable,driver="mscsi6")
1
//if dual
y
1
106-210
y
2
106-210
```

```
y
3
106-210
y
4
106-210
y
5
106-210
y
6
106-210
//end
n
6
203
system (drive=3,disable,driver="mscsi6")
1
//if dual
y
1
106-210
y
2
106-210
y
3
106-210
y
4
106-210
y
5
106-210
y
6
106-210
//end
n
6

//IF format
.
//PAUSE Last chance to abort FORMAT by <BREAK>; <ENTER> to continue
//if dual
mscsif6 :4 (name="hard40m4",mpw="harddisk",#parm#)
//else
mscsif6 :4 (name="hard40a",mpw="harddisk",#parm#)
//end
n
mscsif6 :5 (name="hard40b",mpw="harddisk",#parm#)
n
mscsif6 :2 (name="hard40c",mpw="harddisk",#parm#)
n
mscsif6 :3 (name="hard40d",mpw="harddisk",#parm#)
n
backup :0 :4 (s,i)
swap :0 :4
swap :1 :5
backup :5 :0
```

```
//FLASH 20 From DOS Ready, type the following command line:
. From DOS Ready, type the following command line:
. SYSGEN,DRIVE=4
.
//end
//exit
. HD20INIT/JCL Version 1.0.0 - 09/15/89
. for Model 4 MSCSI installation of 20meg drive
//if noverify
//assign parm="nostop,verify=n"
//else
//assign parm="nostop"
//end
.
. Use: DO HD20INIT (FORMAT,DUAL,NOVERIFY)
. Specify FORMAT to add high-level system information
. Specify DUAL for Model 4 & 3 installation
. Specify NOVERIFY if you don't want MSCSIF/CMD to verify
.
//PAUSE Press <ENTER> to begin the installation
. Bring up the system drive partition of hard drive
system (drive=4,disable,driver="mscsi6")
1
n
4
//if dual
77
system (drive=5,disable,driver="mscsi6")
1
y
1
78-154
y
2
78-154
y
3
78-154
y
4
78-154
//else
153
system (drive=5,disable,driver="mscsi6")
1
//end
n
4
154
system (drive=2,disable,driver="mscsi6")
1
//if dual
y
1
78-154
y
2
78-154
y
3
```

```
78-154
y
4
78-154
//end
n
4
153
system (drive=3,disable,driver="mscsi6")
1
//if dual
y
1
78-154
y
2
78-154
y
3
78-154
y
4
78-154
//end
n
4

//IF format
.
//PAUSE Last chance to abort FORMAT by <BREAK>; <ENTER> to continue
//if dual
mscsif6 :4 (name="hard20m4",mpw="harddisk",#parm#)
//else
mscsif6 :4 (name="hard20a",mpw="harddisk",#parm#)
//end
n
mscsif6 :5 (name="hard20b",mpw="harddisk",#parm#)
n
mscsif6 :2 (name="hard20c",mpw="harddisk",#parm#)
n
mscsif6 :3 (name="hard20d",mpw="harddisk",#parm#)
n
backup :0 :4 (s,i)
swap :0 :4
swap :1 :5
backup :5 :0
//FLASH 20 From DOS Ready, type the following command line:
. From DOS Ready, type the following command line:
. SYSGEN,DRIVE=4
.
//end
//exit
```

FINAL PASS DRIVE UTS REVISION - 2.7

TEST SEQ. 225-0 SERIAL NO. 63894068 NET NO. OFF DCU NO. 42 SYS NO. 33
START DATE 8/10/89 TIME 09:25 STOP DATE 8/10/89 TIME 10:45

DATA HEADS	DATA CYLS	WRITE PRECOMP CYLS	REDUCE WRT CUR	PARK CYL	STEP PULSE RANGE
0-3	0-614	256-614	N/A	670	5-200US

SELECTS AVAILABLE 1 2 3 4 PASS
DE-SELECT PASS
INDEX PASS
TRACK 0 PASS
FULL CYLINDER PASS
STEP

NOM	LOW	HIGH	
3MS	3MS	3MS	PASS

START TIME

NOM	LOW	HIGH	REV	
5.6SEC	6.3SEC	5.2SEC	5.7SEC	PASS

OUTSIDE MARGIN

HEAD 0	HEAD 1	HEAD 2	HEAD 3	
36NS	36NS	34NS	36NS	LATE
42NS	40NS	38NS	40NS	EARLY
78NS	76NS	74NS	76NS	TOTAL
PASS	PASS	PASS	PASS	

INSIDE MARGIN

HEAD 0	HEAD 1	HEAD 2	HEAD 3	
34NS	34NS	26NS	32NS	LATE
32NS	32NS	24NS	30NS	EARLY
66NS	66NS	50NS	62NS	TOTAL
PASS	PASS	PASS	PASS	

SEEK TEST PASS
FORMAT PASS
SINGLE TRACK ACCESS
20.2MS PASS
ROTATIONAL SPEED

NOM	LOW	HIGH	
3600.4RPM	3600RPM	3600RPM	PASS

INDEX PASS
FORMAT PASS
READ/WRITE TEST PASS
TOTAL HARD ERRORS = 0
TOTAL SOFT ERRORS = 0
HARD ERROR MAP :

HD	CYL	MFM BPI	HITS

SEQUENCE REVISION 1.06