WANG OIS

DISK OPERATING SYSTEM

SLAVE PROCESSOR INTERFACE SPECIFICATION

OS-6

DOS Release 9.5
R&D Revision -- 15.0

Jim McCarthy
Priscilla McGan
February 4, 1987

1st Edition -- 1985

Disclaimer of Warranties
and Limitation of Liabilities

The staff of Wang Laboratories, Inc., has taken due care in preparing this manual; however, nothing contained herein modifies or alters in any way the standard terms and conditions of the Wang purchase, lease, or license agreement by which this software package was acquired, nor increases in any way Wang's liability to the customer. In no event shall Wang Laboratories, Inc., or its subsidiaries be liable for incidental or consequential damages in connection with or arising from the use of the software package, the accompanying manual, or any related materials.

NOTICE:

All Wang Program Products are licensed to customers in accordance with the terms and conditions of the Wang Laboratories, Inc. Standard Program Products License; no ownership of Wang Software is transferred and any use beyond the terms of the aforesaid License, without the written authorization of Wang Laboratories, Inc., is prohibited.

PREFACE

This manual describes the communciations interface between the OIS disk operating system and a program running in a slave processor. It describes the functions and features available to the slave program and the means by which the slave program accesses those functions.

The features described in this manual reflect relesae 9.5 of the OIS disk operating system.

Intended Audience

This manual is intended for use by systems and applications programmers wanting to write software packages that run in OIS slave processors (particularly OIS workstations) under the OIS disk operating system.

The reader should have a working knowledge of either Intel 8080 or Zilog Z80 assembly language or of Intel's PL/M80 programming language. No previous knowledge of the OIS environment is required.

Structure of this Manual

This manual has nine chapters:

- Chapter 1 presents an overview of the OIS environment.
- Chapter 2 describes the reserved portion of slave memory.
- Chapter 3 discusses the general concepts involved in issuing requests to the operating system.
- Chapter 4 discusses system disk file operations.
- Chapter 5 discusses direct control of system devices.
- Chapter 6 discusses external file sources.
- Chapter 7 discusses slave processor startup and termination.
- Chapter 8 discusses local disk file operations.
- Chapter 9 discusses miscellaneous requests to the operating system.
- Chapter 10 provides tables of nonimmediate requests and their error codes.
- Chapter 11 describes the Request Logger Utility.

Chapters 4 through 8 are structured such that general concepts are discussed first; the format and details of related requests to the operating system are given in the last section of each chapter.

Conventions

Throughout this manual, the various OIS master processor models are segregated into three categories:

- The Model 30 master processor includes models OIS-125 and 130.
- The Model 40 master processor includes models OIS-105, 115-1, 115-2, 125A, 130A, 140, 145, 40, 50, 60, and 70.
- The EM-OIS master processor includes models EM-OIS-105, 115-1, 115-2, 125A, 130A, 140, and 145.

Hexadecimal values are represented throughout this manual in the form x'yyyy', where yyyy is the value.

Related Documents

For more detailed information on the concepts discussed in this manual, refer to the documents listed below.

OIS Disk Operating System Text File Format

OIS Disk Operating System Queue Files

OIS Disk Operating System External File Source Slave Interface Specification

CONTENTS

CONTENTS (Continued)

CHAPTER 1
INTRODUCTION

This document describes the communication interface between a slave processor and the master processor under the OIS Disk Operating System. It outlines the mechanisms and conventions used by the slave program to access the operating system. In addition, related software is also described, Local I/O and the Request Logger. Further documentation on these software packages is available in OS-80, Local I/O Interface Specification and OS-75, Request Logger.

The following definitions apply throughout the document:

SECTOR
A 256-byte unit of data. All external storage is manipulated in terms of sectors.

DISK
A physical storage medium capable of containing a finite number of sectors.

VOLUME
A finite collection of sectors residing on the same disk.

DRIVE
A physical mechanism that controls one and only one disk.

DISK UNIT
A physical mechanism capable of controlling one or more drives.

SYSTEM RESIDENCE VOLUME
The volume from which the system was IPLed and on which all system programs reside. The name of this volume is the name of the system.

FILE
A logically ordered collection of sectors.

FILE-RELATIVE SECTOR NUMBER (FRSN)
A 3-byte binary number (low-order byte first) that identifies one sector of a file. The FRSN of a sector defines the position of that sector in the file relative to the first sector of the file. The first sector of a file has an FRSN of 0.

FILE SEGMENT

Any logically contiguous group of sectors within a file. (A "contiguous group of sectors" is a group of sectors whose file-relative sector numbers are contiguous integers.)

REFERENCE NUMBER (RN)

A unique 1-byte number assigned to each slave or drive open for direct control and to each open file. The RN is used to identify the drive, slave, or file in all nonimmediate requests to the master. RNs are always in the range of x'01'-x'EF'.

EXTERNAL FILE SOURCE (XFS)

A slave that supplies files to the master. A slave can supply files to the master by using the Define File Source Connection request and by following the rules and protocol outlined in the DOS External File Source Slave Interface Specification.) (The supplying slave's source of files is irrelevant.)

VOLUME-RELATIVE SECTOR NUMBER (VRSN)

A 3-byte binary number (low-order byte first) that identifies one sector of a volume. The VRSN of a sector defines the position of the sector on the volume relative to the first sector of the volume. The first sector on a volume has a VRSN of 0.

CHAPTER 2
SLAVE MEMORY

The first page of a slave processor's memory (locations x'0000' through x'00FF') is typically reserved for use as a general-purpose control area. Some of these locations are required and/or maintained by the master, while others are used (by convention) for transfer of control and communication between slave programs. The remainder of the slave's memory is available for use by slave programs.

The control area is organized as shown below. Each field is discussed separately in the following sections.

Address (hex)	Contents
00-02	Keyboard interrupt routine vector
03-0F	Slave/master communication area (SCA)
10-1F	Global variables
20-2F	System utility variables
30-BF	Slave service routines
C0-FF	Slave program's processor stack

2.1 SLAVE/MASTER COMMUNICATION AREA (SCA)

Slave programs access the features of the operating system by means of a request and acknowledgment convention: The slave makes a request of the master, and the master acknowledges the request. These requests and acknowledgments are passed between the slave and master processors through an area in each slave's memory called the slave/master communication area (SCA). The SCA occupies bytes x'03' through x'0F' of the slave's address space. The SCA is organized as follows:

Address (hex)	Contents
03	Unit number of slave
04-06	Function area

Offset	Contents
00	Function code
01-02	Control block/data pointer (low-order byte first)

Address (hex)	Contents
07	Type code of slave
08-0F	Logon user ID (Model 40 and EM-OIS only)

Byte 3 always contains the slave's unit number (as set by the master when the slave is IPLed). Byte 7 always contains the slave's type code (also set by the master).

Bytes 4-6 are the default function area. Byte 4 contains the request function code, and bytes 5-6 contain a memory pointer (low-order byte first). The function area can be moved to any location in slave memory with the Move Function Area request (Model 40 and EM-OIS only). Once this request is made, the function area is set back to locations x'04'-x'06' only under one of the following circumstances:

- The slave issues another Move Function Area request with an address of x'04'.
- The master performs a Terminate request for the slave.
- The master detects transmit errors when communicating with the slave.
- The slave is opened for direct control (refer to Section 5.2).
- The slave is opened for direct control, and the king requests the master to stop polling the slave (refer to Section 5.2.3).

If no request is pending in the SCA, the function code byte contains a zero. When the slave processor wants to issue a request to the master, it puts a pointer to a control block or data area (depending on the type of request) in bytes x'05'-x'06' and a nonzero request code in byte x'04'. The master acknowledges receipt and processing of the request by setting the function code back to zero. This acknowledgment takes an indefinite amount of time, depending on the request processing time; the slave must not enter another request until the function code is set back to zero.

2.2 GLOBAL VARIABLES

The 16-byte area located from x'10' through x'1F' is used for global communication variables between various system-provided slave programs. This area should not be changed by a nonsystem program unless that program intends to end processing with a Terminate request.

By convention, the global variable area is used as follows:

Address (hex)	Contents
10	Internal menu number of last menu displayed on screen (workstations only)
11	Number of high-memory pages currently occupied by a debugger or by the local I/O package (refer to Chapter 8)
12	Language code
13	Batch handler flag
14	Character set (Model 40 and EM-OIS only)
15	Keyboard code (Model 40 and EM-OIS only)
16-17	(Reserved)
18-1B	Input text file descriptor
1C-1F	Output text file descriptor

2.3 SYSTEM UTILITY VARIABLES

The 16-byte area located from x'20' through x'2F' is used by certain system utility programs for variable space. This area should not be used by slave programs unless the program plans to exit to the operating system via a Terminate request.

2.4 SLAVE SERVICE ROUTINES

The area of memory from x'30' through x'BF' contains two slave service subroutines that are invoked by the RST 6*8 and RST 7*8 instructions. These routines are described in Section 7.3. If a slave program does not intend to use these service routines, it can overwrite this area of memory for its own purposes.

2.5 SLAVE PROGRAM'S PROCESSOR STACK

Memory locations x'C0' through x'FF' are generally reserved for the slave program's processor stack. When a slave program is IPLed or loaded by a service routine, the processor stack pointer register is set to x'0100', pointing to this area. If the program changes the stack pointer register, locating the stack at a different location, this area can be used for anything the slave program wants.

This area is also used for program communications. When one program transfers control to another program, it puts a file name in this area, starting at x'C0'. When the called program finishes processing, it passes control to the named file, using a RST 7*8 (refer to Section 7.3). If the called program uses this area for its stack, it must preserve the file name.

CHAPTER 3
REQUEST PROCESSING

3.1 TYPES OF REQUESTS

There are two types of requests that can be made to the master: immediate requests and nonimmediate requests.

Immediate requests include those functions that require little or no I/O activity by the master. They are complete at the time the master acknowledges the request by resetting the SCA function code to zero. Nonimmediate requests, on the other hand, include those functions that do require I/O activity. They are not complete at the time the master resets the function code.

All nonimmediate requests are issued with an SCA function code of 1. The control block/data pointer within the function area contains the address of a 16-byte area called the request control block (RCB). The RCB contains further information about the request, along with a status byte used by the master at the time the request completes. (The RCB is described in Section 3.2.)

Function codes x'02' through x'3F' are reserved for immediate requests. The control block/data pointer contains a pointer to a data area; the contents of the data area depends on the function requested.

NOTE

The Model 30 requires this data area to be in the low-order 32K bytes of memory. The Model 40 and the EM-OIS, on the other hand, permit it to be anywhere below 64K bytes (but not in screen memory).

If an invalid or unassigned function code is put in the SCA, the master acknowledges the request immediately but takes no other action.

3.2 Request Control Block (RCB)

When a slave issues a nonimmediate request to the master (by loading a function code of 1 into the SCA function code byte), the control block/data pointer field in the SCA must contain a pointer to a 16-byte data area in the slave's memory. This area is known as the request control block (RCB).

The RCBs for all nonimmediate requests have the same basic format.

Offset (hex)	Contents
00	Request code
01	Completion status code
02-0F	Request-specific data

Before issuing a request to the master, the slave program must

- Load the request code with one of the permissible values
- Load the status code with zero
- Load other fields in the RCB as required by the specified request code

NOTE

The user should always define each RCB to be the full 16 bytes and set all bytes that are not used by a particular request to x'00'. This permits easy future expansion and modification of existing requests.

When request processing is completed, the master puts a nonzero value into the status byte. For all requests, the value x'80' means the request completed successfully; the value x'08' means a permanent hardware error occurred during request processing. In this case, byte 9 of the RCB contains the hardware error code. These error codes are defined as follows:

Code (hex)	Meaning
00	No physical damage (Model 40 and EM-OIS only)
01	Drive not ready
02	Disk is write protected
03	CRC error
04	Disk format error
05	Equipment malfunction error
06	Parity error (reading slave memory)
07	Datalink error (writing slave memory)
08	Programming error
09	Drive dropped ready (during operation)
0A	Slave not available
0B	Slave dropped power (during operation)
0C	Write check compare error (Model 40 and EM-OIS only)
0D-7F	Reserved for the operating system
80-FF	Not used

Any status code other than x'80' or x'08' indicates an error or special condition specific to the particular type of request.

Various fields in the RCB may be filled in or changed by the master, depending on the type of request and the outcome of the requested operation. In general, a field is left intact if it is not explicitly used to return information.

3.3 EXTERNAL PROCESSING OF REQUESTS

(Model 40 and EM-OIS only)

The local master (directly connected to the slave by the physical OIS datalink cable) usually processes requests presented to it. There are exceptions to this, however. In a WISE network and in the presence of other external file source (XFS) slaves, requests are sometimes processed external to the local master. (The local master detects XFS slaves when they identify themselves with the Define File Source Connection request. Refer to Chapter 6 for a discussion of XFS slaves.)

Another circumstance under which requests can be processed external to the local master is when a slave is logically attached (via the Attach request) to another system.

3.3.1 Attach to External Systems

Typically a slave is attached (both physically and logically) to its local master. Using the Attach request, however, a slave can be logically connected to the system of its choice. There are several possible results of an Attach request.

- A null system name causes the slave to be attached to the local system.

- A namestring pointer of zero (i.e., no system name provided) has the same effect as a null namestring.

- An illegal character in the system name causes the request to fail but does not affect the attach status of the slave. (That is, it remains logically attached to the system to which it was attached before the request.)

- A syntactically valid system name allows the request to succeed and the slave becomes attached to the system specified, even if the named system does not exist.

- A namestring containing just a semicolon (e.g., /;/) always specifies the local system.

The master stores the name of the system to which each slave is logically attached. Then, if no system name is specified in a request that accepts one, the master uses the name of the logical system as a default. The following requests accept system names:

- Assign Password
- Get File Name
- Open
- Opencat
- Open DC
- Open DCS

The master also automatically directs certain requests (that do not expect a system name) to the slave's logical system master. These requests are as follows:

- Get Drive List
- Get Drive Vector
- Get Slave List
- Get Slave Vector
- Get System Configuration
- Reserve Drive
- Release Drive
- Send Signal

The Attach request itself is always serviced by the local master, and is independent of the system to which the slave is currently attached.

Once a slave has been attached to a system, there are six situations under which the master automatically reattaches the slave to the local system.

- When the master detects that the slave has powered down.
- When the master detects transmission errors from the slave.
- When the master is IPLed. In this case, all slaves are attached to the local system.
- When the slave is opened for direct control.
- When the slave is open under direct control and issues a Terminate request.
- When the slave is under direct control and its king requests the master to stop polling it.

NOTE

Refer to Chapter 5 for a discussion of direct control of slaves and system devices.

CHAPTER 4
SYSTEM FILE OPERATIONS

The OIS Disk Operating System provides a set of requests for use in accessing and manipulating data files on disk. The first portion of this chapter describes the general concepts involved in using these requests. The final section, Section 4.6, describes their formats.

4.1 OPENING FILES

Before a disk file can be manipulated in any way, an Open request must be issued for the file. When issuing this request, the slave program must provide a "namestring" and an "access code". An Open request can be used to open either a new or existing file.

If the Open is for an existing file, the entire file name must be specified in the namestring. If a new file is being opened, a null namestring or a namestring consisting of only a system and/or volume name must be specified. The new file is left unnamed until a Catalog request, discussed in Section 4.2, is issued to name the file. An unnamed file is deleted when it is closed.

Upon completion of the Open request, the master returns the following information to the caller:

- A status code
- The file reference number (RN)
- The file block size (0 for a new file)
- The bytes-in-last-sector value (0 for a new file)
- The end-of-file pointer (0 for a new file)
- The file prologue sector indicator

4.1.1 Namestrings

Many file operations require the user to specify a "namestring" with the operation request. A namestring is a string that identifies a file or group of files by combining a system name, a volume name, node names and a password. The format of a namestiring is as follows:

```
dl [sys;] [[vpw=] vol:] [[[pw =] ntn.] ... [pw=] tn] dl

where:    []       (matched pairs of brackets) indicate optional items
          ...      indicate the preceding bracketed term may be repeated
          dl       is any arbitrary delimiter character (except x'00') not
contained in the rest of the string
          sys      is a system name
          vpw      is a volume password
          vol      is a volume name
          pw       is a node password
          ntn      is a nonterminal node name
          tn       is a terminal node name
          :        is an ASCII colon (x'3A')
          ;        is an ASCII semicolonx ('3B')
          =        is an ASCII equal sign (x'3D')
          .        is an ASCII period (x'2E')
```

The naming elements of a namestring (i.e., system name, volume name, node names, and passwords) can contain letters and digits only.

```
x'30'-x'39'  (0-9)
x'41'-x'5A'  (A-Z)
x'61'-x'7A'  (a-z)
```

The following rules also apply to the formation of namestrings:

- There can be only one equal sign (and therefore only one password) in the namestring. It can be on any file node name or on the volume name.

- There can be no more than five nonterminal node names (ntn.) in the namestring.

- ASCII null characters (x'00') can be embedded anywhere in the namestring; they are completely ignored.

- Namestrings can be no longer than 120(10) bytes, including delimiters and embedded nulls.

The following subsections describe each of the naming elements of a namestring.

System Name
(Model 40 and EM-OIS only)

A system name is a string of from one to eight characters that uniquely identifies a supplier of files. The system name of an OIS system is the volume name of the system residence volume. If a system name is not provided with a request that accepts one, the logical (attached) system is assumed. A semicolon without a system name (e.g., /;/) always means the local system.

Volume Name

A volume name is a string of from one to eight characters that identifies a source of files (usually a disk). The name is permanently recorded on the disk when the volume is initialized. The name of the volume on which a file resides must be provided when the file is opened. If a volume name is not explicitly specified by the requesting slave, the name of the system residence volume is used.

File Name

A file name consists of a group of "node" names, concatenated by an ASCII period ("." = x'2E'), each consisting of a string of from one to eight characters. The maximum number of nodes in a file name is determined at system release time. The absolute maximum is six nodes.

The last node of the file name is called the "terminal" node. Other nodes are called "nonterminal" nodes.

The following special node names are permitted (refer to Section 4.2.2):

x'3C' (ASCII less than)
x'3E' (ASCII greater than)
x'3F' (ASCII question mark)

Password

A password is string of from one to eight characters. It is prefixed to a node name or a volume name (but not a system name) with an ASCII equal sign (x'3D').

4.1.2 Access Code

The access code is used to tell the master what type of operations the slave intends to perform on the file being opened, as well as whether other slaves can use the file at the same time.

There are three types of access available.

- Read-only access: The slave will request only read operations on the file.
- Read/write access: The slave will request both read and write operations and must have exclusive control of the file.
- Shared access: The slave will request read and write operations but does not need exclusive control of the file.

Only one slave is permitted access to a file if it is opened for read/write access. Several slaves, however, can open a file for read-only or shared access.

4.1.3 Status Code

The status code is set by the master to tell the slave whether or not the file was opened successfully. It also provides information about any request failure. Possible reasons for a request failure are as follows:

- An existing file cannot be found on the specified volume.
- The volume itself is not mounted.
- The file is already open with a conflicting access code by this or another slave.
- A format error exists in the namestring.
- The password (if specified) does not grant the user the desired access.
- A permanent I/O error occurred opening the file.

4.1.4 Reference Number (RN)

When an open request succeeds, the master supplies a 1-byte reference number (RN) to be used by the slave to refer to the file in all subsequent requests. A reference number ranges from x'01' to x'EF' and is valid until the file is closed. All files open concurrently by the same slave have unique RNs.

4.1.5 Block Size

A file's block size is a 1-byte number recorded with the file name in the disk catalog. It is supplied by the slave when the file is cataloged and is returned by the master when the file is opened. The slave can use this number in any way it wants.

4.1.6 Bytes-in-Last-Sector

A file's bytes-in-last-sector (BLS) value is a 1-byte number recorded with the file name in the disk catalog. It is supplied by the slave when the file is closed after being open for read/write access and is returned by the master when the file is opened. The slave can use this number in any way it wants.

4.1.7 End-Of-File Pointer

A file's end-of-file pointer is the file-relative sector number of the first free sector at the end of the file. It is also the number of sectors in the file. It is returned by the master when the file is opened.

4.1.8 File Prologue Sector

The file prologue sector (FPS) is an optional sector of information recorded with the file on the volume. It is allocated and written by the slave when the file is created. It can be accessed with the Read, Write, Read Lock and Lock requests by specifying a FRSN of x'FFFFFF'. The slave can also examine the FPS using the Opencat or Get File Name requests. The slave can use this sector in any way it wants.

4.1.9 Password Protection

A node in the catalog can have:

- No password
- A write-prevention password
- A write-prevention password and a read-prevention password

Read/write and shared access are prevented by the write-prevention password; read-only access is prevented by the read-prevention password.

If any node of a file name has a password associated with it in the catalog, a password of the same type must be included in the file namestring of the Open request in order to obtain the corresponding type of access. In other words, if any node of a file name has a write-prevention password associated with it in the catalog, the user must know the write-prevention password of at least one of the file node names in order to write the file. Likewise, if any node of the file name has a read-prevention password associated with it, the user must know the read-prevention password of at least one of the file node names in order to read the file.

An entire volume can have a password. The volume password does not prevent access to files, but it does prevent access to the volume via direct control. Also, if the volume password is correctly specified, the slave can access any file on the volume, regardless of the passwords in the catalog.

If a password is specified on a volume name or node name when no password is required, the specified password is ignored.

A system name cannot have a password associated with it.

4.1.10 Reopening a File

At any time that a file is open, the slave can request a Reopen of the file. The slave provides the RN of the file, a new access code (if desired), a new BLS byte (if desired and only if the current access is read/write), and a pointer to a namestring to be filled in by the master (if desired). Using the Reopen request, a slave can do the following:

- Change the type of access with which the file is opened without closing the file.

- Examine the current end-of-file pointer to determine if the file size has changed since the file was opened.

- Examine the current name and block size of the file.

- Update the BLS byte without closing the file.

Changing the access type of the file from read-only access to either read/write or shared access is permitted only if the password requirements for the new access were met when the file was originally opened (refer to Section 4.1.9).

4.2 CATALOGING FILES

Files are listed in the volume catalog. The catalog is organized in a multilevel tree structure, with each node name of a file corresponding to one level in the tree. The catalog of a volume is a file itself, consisting of one record for each node of the tree. These records are chained together within the file, in an arbitrary nonsequential fashion to form the tree structure.

Nodes within the catalog tree are either "nonterminal" nodes (with nodes below them) or "terminal" nodes (with no nodes below them). A given node cannot be both terminal and nonterminal at the same time. (Thus, for example, the file names X.Y and X.Y.Z cannot coexist in the same catalog since Y would be both terminal and nonterminal.)

4.2.1 Naming a File

The Catalog request is used to change the name of a file. The file must be open with read/write access. When issuing the request, the slave must specify a file reference number and a namestring. The master returns a completion status code.

If the Catalog request succeeds, the file is given the name specified. If the file was previously named, it is renamed; if the file was previously unnamed, it is given the new name. If no name is provided with the Catalog request, the file becomes unnamed (and will be deleted when it is closed, unless it is recataloged).

If an unnamed file is cataloged without a new name, it remains unnamed. While this operation is meaningless, it is, nonetheless, legal. Note, however, that a file cannot be cataloged with its own current name. It must first be cataloged with a null name.

When creating the namestring, the special node name characters x'3C' (ASCII "less than"), x'3E' (ASCII "greater than") and x'3F' (ASCII "question mark") can be used to generate unique node names (refer to Section 4.2.2). If the specified namestring contains a volume or system name, they must be for the volume and system on which the file exists.

The naming of files is affected by any passwords on existing nodes within the catalog tree and may require that the user specify a password with the catalog namestring. A single password is permitted and can be specified for any node of the file name or for the entire volume. If any existing node of the catalog file name has a write-prevention password, then a write-prevention password must be correctly specified on some node of the name in order for the catalog operation to succeed. However, if a valid password is specifed for the volume, the catalog request succeeds regardless of any passwords on existing nodes.

Any number of Catalog operations can be performed on a file while it is open.

4.2.2 File Name Generations

Three special characters are available for use in specifying node names in either a Catalog or an Open request namestring.

- ASCII "less than" (x'3C')
- ASCII "greater than" (x'3E')
- ASCII "question mark" (x'3F')

Within a Catalog Request

When one of these special node name characters is used in a Catalog namestring, the master generates a unique name for that node.

- ASCII "less than": The node name created is less than (within the name collating sequence) the lowest node in that level of the catalog tree.

- ASCII "greater than": The node name created is greater than (within the name collating sequence) the highest node in that level of the catalog tree.

- ASCII "question mark": In a Catalog namestring, "question mark" is functionally equivalent to "less than".

These special characters can be used for any number of node names in the namestring.

The master always generates an 8-character node name and attempts to follow a sequential numbering scheme, using only the ASCII characters 0 through 9. However, if it cannot generate a completely numeric node name, it will use upper- and lower-case letters, as allowed in a standard node name.

If the catalog tree already contains the lowest ("00000000") or the highest ("zzzzzzzz") node name, preventing the master from creating a lower or higher name, the catalog request is refused with a "duplicate file name" error code. If the catalog tree contains no nodes at that level, the master creates a node named "50000000".

As mentioned in Section 4.2.1, a password may be required in a Catalog request namestring to permit naming a file. When the special node name characters are used, passwords are still fully functional. However, passwords are never needed on the special characters themselves since using them always results in the creation of new node names.

Within an Open Request

The special node name characters can also be used for any number of node names within an Open request namestring. When used in this context, the special characters have the following meaning:

- ASCII "less than" is used to specify the lowest node name in the catalog at the specified level.

- ASCII "greater than" is used to specify the highest node name in the catalog at the specified level.

- ASCII "question mark" is functionally equivalent to "less than" when the resulting file is available and the password and access requirements of the requesting slave are met. Otherwise, it is used to specify the next greater node at the specified level (following the one specified by "less than") that meets the password and access requirements.

Passwords are fully functional on all nodes in an Open request file name, including nodes specified with the special characters.

If one of the special node name characters is used as the last specified node of a file name namestring, it is assumed to also stand for any number of trailing nodes under that tree level. That is, if the specified name is only a partial namestring and the final node name is not a terminal node, the master uses the special character to repeatedly define each successive node until it reaches a terminal node.

4.2.3 Assigning Passwords to Nodes

The Assign Password request is used to assign passwords to nodes within the catalog. The slave specifies a partial namestring and two password namestrings. The master returns a completion status code.

The partial namestring defines the catalog tree down to the desired level, specifying a single node. The two password strings specify the write-prevention and read-prevention passwords to be assigned to the node.

If the read-prevention password is null, only a write-prevention password is specified. Both strings can be null, indicating that no password is specified. Note, however, that the read-prevention password must be an ordered subset of the write-prevention password, and cannot be specified without a write-prevention password.

An "ordered subset" means that the read-prevention password must contain one or more characters from the write-prevention password, and those characters must be in the same sequence as in the write-prevention password. For example, if the write-prevention password is "BILLYBOB", the read-prevention password could be "ILY", "BBB", or "BILBO".

Assigning passwords is affected by any passwords already assigned to nodes above the specified node and may require that the user specify a password with the partial namestring. A single password is permitted and can be specified for any node of the partial namestring or for the entire volume. If any node above the specified node already has an assigned write-prevention password, then a write-prevention password must be correctly specified on some node of the partial namestring in order for the operation to succeed. However, if a valid password is specifed for the volume, the catalog request succeeds regardless of any passwords on existing nodes.

4.2.4 Reading the Catalog

Occasionally a slave must access the names and attributes of files in a volume's catalog. However, a slave is not allowed to read the catalog directly since the catalog generally contains information (such as passwords) to which a random user should not have access. Therefore, the Opencat request is used to obtain necessary catalog information.

The Opencat request provides the slave with a scratch file on the system residence volume. This scratch file contains information about one subtree of a catalog. The desired subtree is specified in the form of a partial namestring that defines the catalog tree down to a particular node. Thus, the subtree includes all files under the specified node. If the namestring contains a password, it is ignored.

The scratch file produced by the Opencat request contains a series of ordered fixed-length records, one for each node of the selected subtree. Each 32-byte record is called a node descriptor record (NDR) and contains information such as the node name, node type (nonterminal or terminal), block size, end-of-file pointer, and so on. (Note that file-specific information is valid only in the NDRs of terminal nodes.) The last sector of the scratch file is padded (if necessary) with zero-filled records.

The slave can optionally request (instead of or in addition to the NDR file) a second scratch file containing the file prologue sectors (FPS) of all files in the specified subtree. If both files are requested, each record in the NDR file contains the file-relative sector number of the FPS in the second file.

If the namestring provided with the Opencat request specifies a terminal node rather than a nonterminal node (that is, a file rather than a subtree), no scratch file is produced. Instead, a single node descriptor record or file prologue sector is returned in memory.

4.2.5 Other Ways of Obtaining File Names

The Get File Name request permits a slave to step sequentially through the files in a volume's catalog. The request can be used to obtain the name of the file before or after a given file, or in the case where one of the special node name characters is used (refer to Section 4.2.2), it can be used to obtain the complete name of a file.

When making the Get File Name request, the slave supplies a valid file namestring and a direction indicator. The master returns the complete name of an existing file that is either alphanumerically equal to, immediately before, or immediately after the specified file, as determined by the direction indicator. Passwords have no affect in this request; if a password is included in the namestring, it is ignored.

The Get File Name request can also be used to obtain the file prologue sector (FPS) and the node descriptor record (NDR) for the node of the resulting file name.

4.3 READ AND WRITE OPERATIONS

OIS provides a set of I/O requests for use in transferring data between a file and the slave's memory. Before any I/O operations can be performed on a file, the file must be open by the requesting slave with the appropriate access.

4.3.1 Requesting a Data Transfer

To request a data transfer, the slave supplies the file reference number, a memory buffer pointer, a file-relative sector number, and a count of the number of sectors to be transferred. Upon completion of the transfer, the master returns a status code and a count of the actual number of sectors transferred.

Buffer Pointer

The memory buffer pointer specified with an I/O request must begin on a page boundary and be an integral number of pages long. It should be large enough to contain the requested file segment.

File Segment

The file segment is defined by a file-relative sector number and the number of sectors to be transferred.

When the specified file segment is contained entirely within the existing file, processing is generally straightforward. However, when the file segment exists or extends outside the current bounds of the file, the master must do special processing. The actual processing depends on the nature of the request.

Reading at the End of the File:

When the file segment specified in a read request extends beyond the actual end of the file, the master reads only up to the end of file. The returned sector count reflects the actual number of sectors read, and an "end of file encounterd" status code is returned to the slave.

If the specified file segment falls entirely beyond the end of the file, the master returns an "invalid file segment" status code and sets the returned sector count to zero.

Writing at the End of the File:

A new file is always created with a length of zero sectors. As sectors are written to the end of it, the file grows dynamically.

If the file segment to be written overlaps or starts at the end of the existing file, the file size is increased by the number of sectors necessary to complete the request. The returned sector count is set to the number requested, and a "successful" status code is returned to the slave. If there is not enough free space on the volume to increase the file to the desired size, the returned sector count is set to the number of sectors that were successfully written, and a "volume space exhausted" status code is returned to the slave.

If the specified file segment begins beyond the current end of the file, an "invalid file segment" status code is returned to the slave and no sectors are written.

Reading and Writing the File Prologue Sector:

If the file was created with a file prologue sector (FPS), the FPS resides (logically) at sector -1 (x'FFFFFF') of the file and can be read or written by accessing FRSN -1. All the standard rules and restrictions of file I/O apply equally to the FPS.

Note that sector 0 logically follows the FPS, and, therefore, multisector operations beginning with the FPS are perfectly legal.

4.3.2 Locking a File Segment

Even though a file is open for shared read/write access, it cannot be written unless the requesting slave has (temporary) exclusive access to the file segment. The Read Lock and Lock requests are used to "lock" (temporarily) the desired segment.

These requests are issued just like normal Read requests. However, when the request is complete, the specified file segment is locked, and no other slave can access it. If another slave attempts to read or write the file segment, the request is denied and a "lockout" status code is returned to the slave.

The segment remains locked until the completion of the next file operation requested by the locking slave. This automatic unlocking is done regardless of the nature or success of the next request.

When a Read Lock request results in an "end of file encountered" status code, all sectors within the specified segment are locked, even if they were not all read. In this case, any request by another slave to read into or write past the end of file is denied.

Locking operations under different reference numbers are totally independent. Thus, if a slave opens a file with shared access more than once (and, therefore, has more than one reference number for the same file), a segment locked under one RN is not automatically locked under the other RNs.

4.3.4 Locking an Entire File

With complicated data structures, the Read Lock and Lock requests are not always sufficient to permit conflict-free operations in a shared environment. In these instances, it is sometimes necessary for a slave to temporarily lock an entire file. The File Lock and File Unlock requests are used for this operation.

When issuing a File Lock request, the slave can ask for either exclusive access of "shared read" access (Model 40 and EM-OIS only).

- Exclusive access gives the requesting slave total read/write control over the file; other slaves attempting to read or write the file are returned a "lockout" status code.

- A file can be locked with shared read access by any number of slaves. This form of access gives all locking slaves the right to read the file. However, write requests by these slaves, as well as any I/O access by nonlocking slaves, are denied with a "lockout" indication.

A request to exclusively lock a file fails if another slave has any portion of the file exclusively locked (with a Read Lock, Lock, or File Lock request). However, if another slave has the file locked with shared read access, the request partially succeeds; the file is locked but not exclusively and a "waiting for exclusive access" status code is returned. In this case, read operations will work but write operations may fail with a "lockout" status code. To avoid errors on write operations, the slave should continue to issue exclusive File Lock requests until a successful status code is returned.

Locking a file with shared read access (Model 40 and EM-OIS only) fails only if another slave has some portion of the file exclusively locked (with a Read Lock, Lock, or File Lock request). If other slaves have the file locked with shared read access, the request succeeds.

A file remains locked until the locking slave issues a File Unlock or Close request.

4.3.4 File Update Inquiry

(Model 40 and EM-OIS only)

A slave that has one or more files open with shared read/write access frequently needs to know when the file has been updated by other slaves that also have the file(s) open. The file update inquiry feature provides this information.

Everytime a slave successfully issues a Read, Read Lock, Lock or File Lock request for a shared access file, the master returns a "shared file update" flag in the request control block. This flag indicates whether or not the file has been written to by another slave since the last time the requesting slave successfully locked all or part of the file. If the flag is set, indicating the file has been modified, the slave should Read Lock the file before continuing. This ensures that the data in the slave's memory is current.

Note that a write operation sets the update flag only for other slaves who also have the file open for shared read/write access -- not the requesting slave itself. However, if the requesting slave has the file open more than once (and, therefore, has different reference numbers (RNs) for the same file), a write operation under one RN will set the update flag for the other RNs.

4.4 CHANGING THE SIZE OF A FILE

A file is created with a length of zero sectors and grows dynamically as sectors are written to it. File space can also be explicitly allocated or deallocated with the Set EOF request.

The Set EOF request allows the slave to specify a new end-of-file pointer for a file. The file is either expanded or truncated to make the file exactly the size requested. When the file is expanded, there must be enough space on the volume to accommodate the new size. The added sectors contain random garbage, and reading them without first writing them may result in an I/O error.

The Set EOF operation is valid only on files open for exclusive read/write access or on files open for shared access that have been file locked with exclusive access (Model 40 and EM-OIS only).

The Set EOF request cannot be used to deallocate a file prologue sector.

4.5 CLOSING FILES

When a slave no longer needs access to a file, it issues a Close request to tell the master it is no longer using the file.

If the file is unnamed when it is closed, the master deletes if from the volume and makes the volume space it occupied available for other files. The data in the file is lost.

If the file has a name when it is closed, its end-of-file pointer is updated in the volume catalog. If the file was open with read/write access, the bytes-in-last-sector value (provided by the requesting slave) is updated in the volume catalog.

If the file is locked with exclusive access when the Close request is issued, the master sets the exclusive file lock bit. This bit is returned to other slaves on subsequent Open or Reopen requests. It indicates that the previous slave did not unlock the file before releasing it, and the integrity of the file may be suspect.

Once a file is closed, further requests under the reference number assigned to the file when it was opened are illegal and result in an error. However, the same reference number can be assigned to another file during a subsequent open.

4.6 REQUEST FORMATS

4.6.1 Assign Password

The Assign Password request is used to add or remove passwords to a node in the volume catalog. (Refer to Section 4.2.3.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'0F'

04-05 Pointer to partial or complete namestring of node to which the password is to be assigned.

06-07 Pointer to write-prevention password string or zero. A value of zero means no write-prevention password is defined and passwords will be removed from the node.

08-09 Pointer to read-prevention password string or zero. If these bytes are nonzero, they must point to a password string that is an ordered subset of the password specified in bytes 6 and 7. If bytes 6 and 7 are zero, these two bytes must also be zero.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents	
01	Status Code (in hex):	
	80	Operation successful
	01	Named node not found in volume catalog
	03	Volume not found
	04	Namestring format error
	05	Tree structure error
	06	Password missing or not correct for required access
	07	Invalid new password namestring
	08	Permanent I/O Error
	09	Request or request option not supported (Model 40 and EM-OIS only)
	0A	Catalog damaged
	0B	VAU Map damage

Model 40 and EM-OIS only:

0D	System not accessible
0E	System connection broken
0F	Network overload
12	XFS namestring format error
15	Master processor overload

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

4.6.2 Catalog

The Catalog request is used to change the name of a file. (Refer to Section 4.2.1.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'03'
02	RN of file. (Note that the file must be open.)
03	Block size of the file if a namestring is specified in bytes 4 and 5.
04-05	Pointer to namestring or a value of zero. The namestring must be any standard file namestring, as described in Section 4.1.1. If the system name is included, it must match the system name supplied with the Open request. Likewise, if a volume name is included it, must match the volume name supplied on the Open request. A value of zero in these two bytes means a namestring is not specified and the file will be unnamed.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status Code (in hex)
	80 Operation successful
	81 Successful catalog operation impossible, volume space exhausted
	01 RN invalid
	02 File name already exists in catalog
	03 File not open with read/write access
	04 Namestring format error
	05 Tree structure error
	06 Password missing or incorrect for required access
	07 System or volume name incorrect
	08 Permanent I/O error
	09 Request or request option not supported (Model 40 and EM-OIS only)
	0A Catalog damaged
	0B VAU Map damaged

Model 40 and EM-OIS only:

0E	System connection broken
10	XFS RN invalid
12	XFS namestring format error
13	Operation out of sequence
14	XFS operation out of sequence

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

4.6.3 Close

The Close request is used to relinquish access to a file, slave or disk when it is no longer needed. (Refer to Sections 4.5, 5.1.5, and 5.2.5.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'04'
02	RN of file, slave, or disk.
09	Bytes-in-last-sector (only if file was open for read/write).

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status Code (in hex) 80 Operation successful 01 RN invalid 08 Permanent I/O error 09 Request or request option not supported (Model 40 and EM-OIS only) 0A Catalog damaged 0B VAU Map damaged Model 40 and EM-OIS only): 0E System connection broken 10 XFS RN invalid

If an error of x'08', x'0A', x'0B', x'0E', or x'10' occurs during the operation, the file is closed anyway. However, information recorded in the volume catalog (end-of-file pointer, bytes-in-last-sector value, access dates, etc.) may be incorrect.

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

4.6.4 File Lock

The File Lock request is used to temporarily lock an entire file. (Refer to Section 4.3.3.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'0D'
02	RN of file.
0C	Access Code (in hex) 00 - Exclusive 01 - Shared read (Model 40 and EM-OIS only)

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex) 80 Operation successful 81 Entire file is locked, but waiting for exclusive access (Model 40 and EM-OIS only) 01 RN invalid 03 File not open for shared read/write access 05 Lockout 09 Request or request option not supported (Model 40 and EM-OIS only) 0B VAU Map damaged Model 40 and EM-OIS only: 0E System connection broken 10 XFS RN invalid 13 Operation out of sequence 14 XFS operation out of sequence
09	If the status code is x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

0B Shared file update flag (in hex) (Model 40 and EM-OIS only).

00 The file has not been written by another user since the last successful Lock or File Lock by this user.

01 The file has been written by another user since the last successful Lock or File Lock by this user.

02-FF Reserved

0C Indicator Byte (Model 40 and EM-OIS only):

Bit 0 Reserved

Bit 1 Exclusive file lock flag:

0 - File was closed normally.

1 - File was closed while exclusively locked.

Bits 2-7 Reserved

4.6.5 File Unlock

The File Unlock request is used to release a previously locked file. (Refer to Section 4.3.3.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'0E'
02	RN of file.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex) 80 Operation successful 01 RN invalid 03 File not open for shared read/write access 06 File not previously locked 09 Request or request option not supported (Model 40 and EM-OIS only) 0B VAU Map damaged Model 40 and EM-OIS only: 0E System connection broken 10 XFS RN invalid 13 Operation out of sequence 14 XFS operation out of sequence
09	If the status code is x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

4.6.6 Get File Name

The Get File Name request is used to sequentially step (forward or back) through the file names in the volume catalog or to determine the complete name of a file specified using the special node name characters. (Refer to Section 4.2.5.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request code = x'0C'
03	Request option (in hex): 3C (ASCII less than) = Previous name 3D (ASCII equal) = Current name 3E (ASCII greater than) = Next name

04-05 Pointer to input namestring. The namestring must be a standard file namestring, as described in Section 4.1.1. If no system name is specified the default system name is assumed. If no volume name is specified the system residence volume is assumed. Note that a value of zero is not permitted for these two bytes; a namestring must be specified.

06-07 Pointer to buffer to receive namestring returned from the master. It can contain the same value as bytes 4 and 5.

0A-0B Pointer to buffer to receive the node descriptor records of each node of the file named by the master. If the value here is zero, no node descriptor record is returned.

0C Page number of buffer to receive the file prologue sector of the file named by the master. If the value here is zero, the file prologue sector is not returned.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation successful 01 File name not found on volume 02 End of catalog encountered (x'3C' and x'3E' only) 03 Volume not found 04 Namestring format error 05 Tree structure error

07 Master processor overload (Model 40 and EM-OIS only)
08 Permanent I/O error
09 Request or request option not supported
0A Catalog damaged
0B VAU Map damaged
Model 40 and EM-OIS only:
0D System not accessible
0E System connection broken
0F Network overload
12 XFS namestring format error

06-07 The buffer pointed to by these two bytes is filled with the namestring of the current, next, or previous file (if requested). The system name is included if the local master is configured for XFS. The volume name is included if the file is not on the system residence volume. The namestring is always enclosed in quotation marks (").

08 File prologue sector flag for the resulting file:
00 File has no file prologue sector
01 File has a file prologue sector

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code is x'80', this byte contains the number of node descriptor records returned to the buffer pointed to by bytes A and B.

0A-0B If the slave specified a nonzero buffer pointer in these two bytes, the buffer is filled with the node descriptor records of each node of the file named by the master. (Refer to Section 4.6.9 for the details of a node descriptor record.)

0C If the slave specified a nonzero page number in this byte, the file prologue sector of the file named by the master is read into slave memory at the given page.

4.6.7 Lock

The Lock request is used to temporarily lock a file segment. (Refer to Section 4.3.2.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'0B'
02	RN of file.
03	Number of sectors.
06-08	File-relative sector number of the start of the file segment.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation successful 01 RN invalid 03 File not open for shared read/write access 05 Lockout 06 Segment already locked with shared read for this RN (Model 40 and EM-OIS only) 09 Request or request option not supported (Model 40 and EM-OIS only) 0B VAU Map damaged Model 40 and EM-OIS only: 0E System connection broken 10 XFS RN invalid 13 Operation out of sequence 14 XFS operation out of sequence
09	If the status code is x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)
0B	Shared file update flag (in hex) (Model 40 and EM-OIS only): 00 The file has not been written by another user since the last successful Lock or File Lock by this user. 01 The file has been written by another user since the last successful Lock or File Lock by this user. 02-FF Reserved

4.6.8 Open

The Open request is used to make a file available for access. (Refer to Section 4.1.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request code = x'00'
03	Access code 00 Read-only 01 Read/write 02 Shared read/write

04-05 Pointer to namestring specifying the file to be opened. The namestring must be a standard file namestring, as described in Section 4.1.1. If no system name is given the default system name is used; if no volume name is given the system residence volume is assumed. These two bytes can also contain a zero or point to a namestring consisting of only a system or volume name. In this case, a new file will be created and opened.

0A Page number of memory buffer containing the file prologue sector for a new file. If the value here is nonzero, the master allocates a file prologue sector for the file and initializes it with the contents of the specified page. This field is ignored when opening an existing file.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status Code (in hex) 80 Operation successful 01 File not found on volume 02 Access conflict 03 Volume not found 04 Namestring format error 05 Tree structure error 06 Password missing or incorrect for requested access 07 Master processor overload (Model 40 and EM-OIS only) 08 Permanent I/O error 09 Request or request option not supported (Model 40 and EM-OIS only)

0A Catalog damaged
0B VAU Map damaged
0C Insufficient space on volume to create file prologue sector

Model 40 and EM-OIS only:
0D System not accessible
0E System connection broken
0F Network overload
12 XFS Namestring format error

02 RN assigned to the file (valid only if status = x'80').

03 Block size of the (existing) file.

06-08 End-of-file pointer for the (existing) file.

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code = x'80' (success), this byte contains the bytes-in-last-sector of the (exisitng) file.

0B File prologue sector indicator:
00 File does not have a file prologue sector.
01 File has a file prologue sector.

0C Indicator Byte (Model 40 and EM-OIS only):
Bit 0 Reserved
Bit 1 Exclusive file lock flag:
0 - File was closed normally.
1 - File was closed while exclusively locked.
Bit 2 Close during write flag:
0 - File was closed normally.
1 - File was closed during write operation.
Bits 3-7 Reserved

4.6.9 Opencat

The Opencat request is used to read the volume catalog. (Refer to Section 4.2.4.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'05'

03 Size of the primary work buffer (in pages) to be used by the master to build the file of node descriptor records (NDRs). If the value here is zero, no NDR file will be built.

04-05 Pointer to primary work buffer. When the request is made, this buffer contains the full or partial namestring to be processed by the master. If byte 3 is nonzero, the buffer is then used by the master as a work area while building the file of node descriptor blocks; it must be on a page boundary (and on a System 125 or 130, it must reside entirely in the lower 32K of slave memory). If byte 3 is zero (indicating no file is to be built), the buffer is used only to contain the namestring and need not be on a page boundary. (However, on a System 125 or 130, it must still be in the lower 32K of slave memory.)

0A-0B Pointer to secondary work buffer to be used by the master to build the file of prologue sectors (FPSs). It must be on a page boundary and can be anywhere in memory. If byte C is zero, this field is ignored.

0C Size of secondary work buffer (in pages). If the value here is zero, no FPS file will be built.

NOTE

The buffers provided should be as large as possible. Their size directly affects the speed with which the Opencat request completes.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status Code (in hex)
	80 Operation successful
	81 Successful Opencat operation impossible, system residence volume space exhausted
	01 Node not found in volume catalog
	02 Namestring specified a terminal node (i.e., a file). In this case no files are built and data is returned as follows: o The primary work buffer contains the node descriptor record for the named node. o If a file of prologue sectors was requested, the secondary work buffer contain the FPS for the named node.
	03 Volume not found
	04 Namestring format error
	05 Tree structure error
	06 Buffer not on page boundary
	07 Master processor overload
	08 Permanent I/O error
	09 Request or request option not supported (Model 40 and EM-OIS only)
	0A Catalog damaged
	0B VAU Map damaged
	Model 40 and EM-OIS only:
	0D System not accessible
	0E System connection broken
	0F Network overload
	11 XFS buffer not on page boundary
	12 XFS namestring format error
02	Reference number (RN) of the NDR file. (Node descriptor records are described below.)
06-08	End-of-file pointer for the NDR file.
09	If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code = x'80' (success) and file prologue sectors were requested, this byte contains the reference number of the FPS file.

Node descriptor records are 32 bytes long and have the following format:

Bytes (hex)	Contents
00-07	Node name
08	Node Type 0 - Nonterminal 1 - Terminal (this NDR describes a file)
09	Tree level of this node
0A-0E	Date node was last written
0F-13	Date of last access (files only)
14-16	End-of-file pointer (files only)
17-18	Physical VAU number of 1st VAU (files only)
19	Block Size (files only)
1A	Bytes-in-last-sector (files only)
1B	Password Flag: 00 Node has no password. 01 Node has a write-prevention password. 02 Node has a write-prevention password and a read-prevention password.
1C-1D	File prologue sector field (files only): If no FPS file is created: x'0000' File has an FPS. x'FFFF' File has no FPS. If FPS file is created: x'FFFF' File has no FPS. Otherwise Value is the FRSN of the FPS in the FPS file.
1E-1F	Reserved

4.6.10 **Read**

The Read request is used to read a file, slave, or disk. (Refer to Sections 4.3, 5.1.3, and 5.2.2.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'01'
02	RN of file, slave, or disk.
03	Number of sectors or pages.
04-05	Pointer to memory buffer to receive sectors or pages. This buffer must begin on a page boundary.
06-08	For files: The FRSN of the start of the file segment to be read. For disks under direct control; The VRSN of the first sector of the disk to be read. For slaves under direct control: Pointer to the start of slave memory to be read (in bytes 6-7). Byte 8 is unused.
0A	Integral request field. Refer to Section 6.3. (Model 40 and EM-OIS only)

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status Code (in hex) 80 Operation successful, all requested data read 81 Operation successful, less than requested amount of data read (i.e., end-of-file encountered) 01 RN invalid 02 Buffer not on page boundary 04 Invalid file segment specified (out of bounds) 05 Lockout (shared files only) 08 Permanent I/O error 09 Request or request option not supported (Model 40 and EM-OIS only) 0B VAU Map damaged

Model 40 and EM-OIS only:

0E	System connection broken
0F	XFS RN invalid
11	XFS buffer not on page boundary
13	Operation out of sequence (shared files only)
14	XFS operation out of sequence (shared files only)

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code = x'80' or x'81' (success), this byte contains the actual number of sectors or pages read.

0B Shared file update flag (in hex) (Model 40 and EM-OIS only)

00	The file has not been written by another user since the last successful Lock or File Lock by this user.
01	The file has been written by another user since the last successful Lock or File Lock by this user.
02-FF	Reserved

4.6.11 Read Lock

The Read Lock request is used to simultaneously read and lock a file segment. (Refer to Section 4.3.2.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'06'
02	RN of file.
03	Number of sectors.
04-05	Pointer to memory buffer to receive data. This buffer must be on a page boundary.
06-08	File-relative sector number of start of file segment.
0A	Integral request field. Refer to Section 6.3. (Model 40 and EM-OIS only)

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex)
	80 Operation successful, all sectors read
	81 Operation successful, less than requested number of sectors read (i.e., end-of-file encountered)
	01 RN invalid
	02 Buffer not on page boundary
	03 File not open with shared read/write access
	04 Invalid file segment specified (out of bounds)
	05 Lockout
	06 Segment already locked with shared read access for this RN (Model 40 and EM-OIS only)
	08 Permanent I/O error
	09 Request or request option not supported (Model 40 and EM-OIS only)
	0B VAU Map damaged
	Model 40 and EM-OIS only:
	0E System connection broken
	10 XFS RN invalid
	11 XFS buffer not on page boundary
	13 Operation out of sequence
	14 XFS operation out of sequence

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code = x'80' or x'81' (success), this byte contains the actual number of sectors read.

0B Shared file update flag (in hex) (Model 40 and EM-OIS only)

00 The file has not been written by another user since the last successful Lock or File Lock by this user.

01 The file has been written by another user since the last successful Lock or File Lock by this user.

02-FF Reserved

4.6.12 **Reopen**

The Reopen request is used to reopen a file, disk or slave while it is already open, thereby permitting the caller to change the access type or to retrieve current device-specific information. (Refer to Section 4.1.10.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request code = x'08'
02	RN of the file, slave, or disk to be reopened.
03	New access code (in hex): 00 Read-only (files only) 01 Read/write (files only) 02 Shared read/write (files only) FF No change
04-05	Pointer to memory buffer to receive the namestring of the file being reopened. If this value is zero, no namestring is returned. This field is unused for direct control devices.
09	Bytes-in-last-sector. This value is used only if the file is currently open for read/write and the access is being changed.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation successful 01 RN invalid 02 Access conflict -- requested access cannot be obtained 06 Password provided when file was opened was insufficient to allow requested access change 08 Permanent I/O error 09 Request or request option not supported (Model 40 and EM-OIS only) 0A Catalog damaged 0B VAU Map damaged

Model 40 and EM-OIS only:

0E	System connection broken
10	XFS RN invalid
13	Operation out of sequence
14	XFS operation out of sequence

03 Block size currently defined for file; low-order byte of sectors per cylinder for direct control disk.

04-05 If a namestring was requested, the memory buffer pointed to by this values is filled with the namestring. The system name is included if the local master is configured for XFS. The volume name is included if the file is not on the system residence volume. The namestring is always bracketed with quotation marks ("). This pointer is ignored for direct control devices.

06-08 Current end-of-file pointer of the file; disk size (in sectors) for direct control disks.

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code = x'80' or x'81' (success), this byte contains the following information:

For files: Bytes-in-last-sector
For direct control disks: Drive address
For direct control slaves: Unit address

0A For files: If the status code = x'80', this is the new access code.

For direct control disks: x'03'
For direct control slaves: x'04'

0B For files: File prologue sector flag

00	File does not have a file prologue sector
01	File has a file prologue sector

For direct control disks:
High-order byte of sectors per cylinder

0C Indicator byte (Model 40 and EM-OIS only):

Bit 0 Reserved
Bit 1 Exclusive file lock flag:
0 - File was closed normally.
1 - File was closed while exclusively File Locked.
Bits 2-7: Reserved

4.6.13 Set EOF

The Set EOF request is used to change the size of a file by redefining the end-of-file pointer. (Refer to Section 4.4.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'07'
02	RN of file.
06-08	New file-relative sector number of the last sector in the file, plus 1. This values is also the new number of sectors in the file.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex) 80 Operation successful 01 RN invalid 03 File not open with write access 04 Insufficient space on volume to increase file size as requested 05 Lockout (shared files only) (Model 40 and EM-OIS only) 06 Segment not previously locked (shared files only) (Model 40 and EM-OIS only) 08 Permanent I/O error 09 Request or request option not supported (Model 40 and EM-OIS only) 0B VAU Map damaged Model 40 and EM-OIS only: 0E System connection broken 10 XFS RN invalid 13 Operation out of sequence 14 XFS operation out of sequence
09	If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

4.6.14 Write

The Write request is used to transfer data to a file, disk, or slave that is open for access. (Refer to Sections 4.3, 5.1.3, and 5.2.2.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'02'
02	RN of file, slave, or disk.
03	Number of sectors or pages.
04-05	Pointer to memory buffer containing data. Buffer must be on a page boundary.
06-08	For files: FRSN of the start of the file segment to be written. For disks under direct control: VRSN of the first sector of the disk to be written. For slaves under direct control: Pointer to the start of slave memory to be written (in bytes 6-7). Byte 8 is unused.
0A	Integral request field. Refer to Section 6.3. (Model 40 and EM-OIS only)

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex) 80 Operation successful, all data written 81 Operation successful, less than requested amount of data written (i.e., volume space exhausted) 01 RN invalid 02 Buffer not on page boundary 03 File not open with write access 04 Invalid file segment specified (out of bounds) 05 Lockout 06 Segment not previously locked (shared files only) 08 Permanent I/O error 09 Request or request option not supported (Model 40 and EM-OIS only) 0B VAU Map damaged

Model 40 and EM-OIS only:

0E	System connection broken
10	XFS RN invalid
11	XFS buffer not on page boundary
13	Operation out of sequence (shared files only)
14	XFS operation out of sequence (shared files only)

09 If the status code is either x'08' or x'0B' (Model 40 and EM-OIS only), this byte contains a hardware error code. (Refer to Section 3.2 for a list of hardware error codes.) If the status code = x'80' or x'81' (success), this byte contains the actual number of sectors or pages written.

0B Shared file update flag (in hex) (Model 40 and EM-OIS only)

00	The file has not been written by another user since the last successful Lock or File Lock by this user.
01	The file has been written by another user since the last successful Lock or File Lock by this user.
02-FF	Reserved

CHAPTER 5
DIRECT CONTROL OF SYSTEM DEVICES

The OIS Disk Operating System provides a set of requests that slaves can use to directly control system devices. Using these direct control requests, a slave can read, write, and issue commands to both disk drives and other slaves (Model 40 and EM-OIS only). Using these requests, the slave can also override the logical file structure imposed by the standard system file requests.

The first portion of this chapter describes the general concepts involved in using these requests. The final section, Section 5.3, describes the request formats.

5.1 DIRECT CONTROL OF DISKS

Using the direct control feature, slaves can mount, dismount, read and write both DOS and non-DOS volumes.

A DOS volume is a disk that has been initialized (using DOS volume utilities) with a suitable volume label, a volume allocation (VAU) map, and a catalog. In addition, a DOS volume is identified by a special redundancy check number encoded in the label. Therefore, a DOS volume is a disk where the volume label sector of the disk is readable without I/O error, and the format and redundancy check conform to that of a DOS volume label.

5.1.1 Volume Mounting

To mount a volume, the slave must determine the address and state of the drive on which the volume is to be mounted, reserve the drive for its own use, have the operator mount the required volume, and then release the drive. Once the volume is mounted, the slave should re-examine the state of the drive to verify that the proper volume is mounted.

Examining the State of the Drive

To examine the state of a drive, the slave uses the Get Drive List and Get Drive Vector requests.

The Get Drive List request returns a list of all drives on the system, along with the following information for each drive:

- Drive address
- Drive type
- Drive size
- State code

Once the slave has determined (according to size and type) the address of the drive on which the volume should be mounted, it uses the Get Drive Vector request. This request provides specific information about a single drive, including such things as volume name, catalog format level, and so on.

Reserving and Releasing a Drive

The slave reserves a drive for its own use with the Reserve Drive request. Upon successful completion of this request, the master marks the specified drive "reserved" for the requesting slave. In addition, any other drives on the same disk unit are marked "indisposed" on behalf of the requesting slave (unless those drives are already reserved by the slave).

In order for the reserve operation to succeed, the specified drive must be in one of the following states:

- No disk in drive
- Not in use
- Indisposed (on behalf of the requesting slave)

In addition, all other drives controlled by the same disk unit, must be in one of the following states:

- No disk in drive
- Not in use
- Indisposed (on behalf of the requesting slave)
- Reserved (on behalf of the requesting slave)

Once a slave has reserved a drive through the Reserve Drive request, the drive stays reserved until that slave releases it. The drive is released when the slave issues a Release Drive request or a Terminate request, or when the slave is re-IPLed.

NOTE

Reserving a drive does not give the slave access to the data on that disk. It merely provides insurance that the drive is not in use by the master or any other slave.

To release a drive, the slave issues the Release Drive request. If any other drives on the same unit are currently "reserved" for the requesting slave, the master marks the specified drive "indisposed" on behalf of the requesting slave. If all other drives are currently "indisposed" on behalf of the requesting slave, all drives on the unit are marked either "no disk in drive" or "not in use".

When the specified drive is actually released (rather than being marked "indisposed"), the master attempts a logical mount operation to put the drive in the "mounted, not in use" state.

5.1.2 Opening a Disk for Direct Control

Before accessing a disk under direct control, the slave must open the entire disk as if it were a single file. This is done with the Open DC (Open for Direct Control) request. In place of a file name, the slave provides a 1-byte drive address and, optionally, the name of the system to which the disk is physically attached. If the disk is a DOS volume with a volume password recorded in its volume label, the slave must also supply the password. The master returns a reference number, the number of sectors on this disk, and the cylinder size (in sectors).

Once a disk is open for direct control, the slave has exclusive access it.

5.1.3 Reading and Writing Disks under Direct Control

Using the reference number supplied by the master when the disk was opened, the slave can issue standard file I/O requests to access disk data. However, instead of specifying a file-relative sector number to describe the beginning of the data to be transferred, the slave specifies a volume-relative sector number. All other parameters supplied with the requests are the same.

A disk open for direct control is treated as if it were a single file. An "end of file encountered" or "volume space exhausted" status code means the I/O request attempted to go beyond the physical end of the disk.

Read, Write, Reopen, and Close are the only system file operations valid for direct control disks.

5.1.4 Direct I/O Operations to Disk

In addition to using standard file I/O requests, the slave can use the Direct I/O to Disk request to perform any physical operation on the disk (except for Unlock Floppy Door). The master passes the request input parameters (specified by the slave in the request control block) directly to an input/output queue entry (IOQE). The IOQE is then queued to the physical disk handler. When the data transfer is complete, the master returns status information to the slave directly from the IOQE.

5.1.5 Closing a Direct Control Disk

When the slave is done with the disk, it issues a standard Close request, specifying the reference number of the volume. At that point, the master marks the drive "mounted, not in use".

5.2 Direct Control of Slaves

(Model 40 and EM-OIS only)

OIS also permits a slave to have direct control of another slave. Using this feature, the controlling slave (known as the "king") can read and write the memory of the controlled slave (known as the "pawn") as if it were a disk file. The king can also perform certain control functions on the pawn.

5.2.1 Opening a Slave for Direct Control

Before a slave can become a king, it must open the target pawn for direct control using the Open DCS (Open Slave for Direct Control) request. In place of a file name, the potential king provides a 1-byte slave unit number and, optionally, the name of the system to which the slave is physically attached.

The master grants the king direct control of the pawn only if the pawn is currently idle and not already under direct control. (A king can have several pawns at once, but a pawn can have only one king.) The requesting slave can determine the current state of its target pawn by issuing the Get Slave List and Get Slave Vector requests.

If the target pawn is available, the master IPLs the pawn with a program that clears the pawn's memory. This is done as a security measure, preventing the king from having access to any potentially sensitive data in the pawn's memory.

Once the pawn's memory is clear, the master returns a reference number to the king for use in subsequent requests. The king is not told the size of the pawn's memory. (Indeed, the master itself has no access to this information.)

If a king is opened for direct control by another slave (because it has neglected to set its activity code), it loses its pawns. The pawns are re-IPLed.

5.2.2 Reading and Writing Slaves under Direct Control

Once a slave is open for direct control, the king can use standard file I/O requests to transfer data to and from the pawn's memory as if it was a disk file. However, in place of a file-relative sector number and a sector count, the king specifies a memory address and a page count to describe the data to be transferred.

Note that since the master does not know the size of the pawn's memory, I/O transfers to nonexistent pawn memory can cause unpredictable results.

Read, Write, Reopen, and Close are the only system file operations valid for direct control slaves.

5.2.3 Direct I/O Operations to Slaves

In addition to the standard file I/O requests, the king can use the Direct I/O to Slave request to perform the following I/O and control functions:

- Read 1 to 255 bytes from the slave's memory
- Write 1 to 255 bytes to the slave's memory
- Read the slave's status
- Reset the slave
- Start slave execution at a specified memory address
- Suspend master polling of the slave
- Resume master polling of the slave

Polling Slaves under Direct Control

When a slave is opened for direct control, the master immediately stops polling it for requests and does not resume polling until it is instructed to do so by the king or until the king closes the pawn. The master does not do any form of initialization on the pawn before it starts polling. Therefore, the king must ensure that the pawn's slave/master communications area (SCA) is reset to some reasonable state (typically zero) before polling is resumed.

Note also that once polling is resumed, the pawn can become a king itself.

5.2.4 Slave Termination

When a pawn terminates or powers down, it remains under direct control but is not polled.

When a king terminates or powers down, its resources are returned to the system. In particular, each of its pawns is closed and released from direct control.

5.2.5 Closing a Direct Control Slave

A slave open for direct control is released with the standard Close request.

If a pawn is being polled when it is closed by its king, the master continues polling. The program currently running in the pawn is permitted to continue and is unaffected by the close; it retains control of its open files and other system resources (including its own pawns).

If the pawn is not being polled when it is closed by its king, the master IPLs the pawn.

5.3 REQUEST FORMATS

5.3.1 Direct I/O to Disk

The Direct I/O to Disk request is used to perform direct physical I/O to a disk open for direct control. (Refer to Section 5.1.4.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'0A'
02	RN of disk.
03	Sector count (passed to the IOQE).
04	Request code, in hex (passed to the IOQE): C0 Read sectors C1 Write sectors C2 Check sectors (Model 40 and EM-OIS only) C3 Format cylinder C4 Seek cylinder specified by volume-relative sector number (VRSN) (Model 40 and EM-OIS only) C5 Restore (return to cylinder zero) (Model 40 and EM-OIS only) C6 Reserved for use by Local I/O
05	Page number (passed to the IOQE).
06-08	Volume-relative sector number (passed to the IOQE).
0B	Reserved for use by Local I/O

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation completed 01 RN invalid 03 RN is not for a direct controlled disk Model 40 and EM-OIS only: 09 Request or request option not supported 0E System connection broken 10 XFS RN invalid

03	I/O queue retry count.
06-08	Reserved for use by Local I/O
09	I/O queue status code for completed operation (in hex): 80 Operation completed successfully 01 Operator intervention required 02 Write protection error 03 CRC error on read 04 Disk format error 05 Equipment malfunction 06 Parity error reading slave memory 07 Datalink error writing slave memory 08 Programming error 09 Device dropped ready during operation 0A Addressed slave not available 0B Slave dropped power during operation 0C Write check error (Model 40 and EM-OIS only)
0A	Reserved for use by Local I/O
0C	Cause of last I/O queue retry. (Model 40 and EM-OIS only)

5.3.2 Direct I/O to Slave

Model 40 and EM-OIS only

The Direct I/O to Slave request is used to read or write a controlled slave's memory or to perform other control functions on the slave. (Refer to Section 5.2.3.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'0A'
02	RN of slave.
03	Length of string to read or write (x'01' to x'FF')
04	Request code (in hex): Code Function 00 Read a string of 1 to 255 bytes from slave memory 01 Write a string of 1 to 255 bytes to slave memory 02 Read the slave's status 03 Reset the slave 04 Start program execution a the specified slave address 05 Suspend master polling of the slave 06 Resume master polling of the slave
06-07	Pawn address: Read or write: address of data to be transferred Start execution: address at which to start execution (must be on a page boundary)
08-09	King address: Read or write: address of data to be transferred

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation successful 01 RN invalid 03 RN is not for a direct controlled slave 08 Permanent I/O error 09 Request or request option not supported 0E System connection broken 10 XFS RN invalid

03 If the status code in byte 1 equals x'08' and the hardware error code in byte 9 equals x'06' or x'07', this byte contains the number of transmission errors.

08 If the request code was x'02', this byte contains the slave's status.

09 If the status code in byte 1 equals x'08', this byte contains the hardware error code:

06	Parity error reading slave memory
07	Datalink error writing slave memory
08	Programming error

5.3.3 Get Drive List

The Get Drive List request is used to obtain a table describing the disk drives currently supported by the system. (Refer to Section 5.1.1.)

Request type: Immediate

Function code: x'05'
Data pointer: Pointer to data area that is to receive list

If the request is unsuccessful, the first byte of the receiving area is set to x'FF' and the second byte is loaded with an error code (Model 40 and EM-OIS only). The error codes are as follows:

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, the receiving area is filled with 2-byte entries, each describing one drive supported by the system. The first byte of each entry is the drive address. The current list of address assignments is as follows:

Address (hex)	Decoded Address r s s s u u p p	Drive Type	Removable/Fixed Volume	Formatted Capacity (MB)
01	0 0 0 0 0 0 0 1	First Floppy	Removable	0.25
02	0 0 0 0 0 0 1 0	Second Floppy	Removable	0.25
04	0 0 0 0 0 1 0 0	First Hawk-5	Fixed	
84	1 0 0 0 0 1 0 0	First Hawk-5	Removable	5.
06	0 0 0 0 0 1 1 0	First Hawk-2	Fixed	
86	1 0 0 0 0 1 1 0	First Hawk-2	Removable	2.5
08	0 0 0 0 1 0 0 0	Second Hawk-5	Fixed	
88	1 0 0 0 1 0 0 0	Second Hawk-5	Removable	5.
0A	0 0 0 0 1 0 1 0	Second Hawk-2	Fixed	
8A	1 0 0 0 1 0 1 0	Second Hawk-2	Removable	2.5
Model 40 and EM-OIS only:				
14	0 0 0 1 0 1 0 0	Winchester-3	Fixed	
24	0 0 1 0 0 1 0 0	Winchester-4	Fixed	
34	0 0 1 1 0 1 0 0	Winchester-8	Fixed	
44	0 1 0 0 0 1 0 0	Winchester-20	Fixed	
54	0 1 0 1 0 1 0 0	Winchester-40	Fixed	
58	0 0 1 1 1 0 1 0	Winchester-40(8)	Fixed	33.5
1n	0 0 0 1 0 0 - -	SMD-80	Removable	67.
2n	0 0 1 0 0 0 - -	SMD-300	Removable	275.
3n	0 0 1 1 0 0 - -	CMD-32	Fixed	
Bn	1 0 1 1 0 0 - -	CMD-32	Removable	13.
4n	0 1 0 0 0 0 - -	CMD-64	Fixed	
Cn	1 1 0 0 0 0 - -	CMD-64	Removable	13.

Address (hex)	Decoded Address r s s s u u p p	Drive Type	Removable/Fixed Volume	Formatted Capacity (MB)
Model 40 and EM-OIS only (cont.):				
5n	0 1 0 1 0 0 - -	CMD-96	Fixed	67
Dn	1 1 0 1 0 0 - -	CMD-96	Removable	13
6n		NEC-85	Fixed	67
7n		NEC-160	Fixed	134
9n		FSD-344	Fixed	344
An		FMD-675	Fixed	675
Model 50:				
03	0 0 0 0 0 0 1 1	Mini-Floppy	Removable	0.36
18	0 0 0 1 1 0 0 0	Mini-Win-80	Fixed	5.
28	0 0 1 0 1 0 0 0	Mini-Win-42	Fixed	5.
38	0 0 1 1 1 0 0 0	Mini-Win-10	Fixed	10.
48	0 0 1 1 0 0 0 0	Mini-Win-30	Fixed	33.6

The above table is decoded as follows:

r = 0 means "not removable", therefore fixed; r = 1 means "removable"

sss gives size indication:
if sss=000, then the address is on a 10MB floppy controller or a mini-floppy controller.
if sss is not 000, then it is an SMD/CMD or Winchester:
(higher values correspond to larger sizes)

uu gives use indication:
on 10MB floppy controller:
if uu=00, it is a floppy disk or mini-floppy
if uu=01 or uu=10, then uu is the Hawk unit number
otherwise:
if uu=00, then it is an SMD/CMD address
if uu=01, then it is a Winchester address
if uu=10, then it is a Mini-Winchester address

pp gives the disk particular or plug information:
if sss=000 and uu=00:
if pp=01 or 02, it is the floppy diskette unit number
if pp=03, it is a mini-floppy
if sss=000 and (uu=01 or uu=10):
pp indicates the Hawk size:
if pp=00, then the Hawk is 5.0 MB
if pp=10, then the Hawk is 2.5 MB
if sss not=000 and uu=00:
pp=n the SMD/CMD plug (unit) number
if sss not=000 and uu=01:
pp=00 for all Winchester unit numbers
if sss not=000 and uu=10:
pp=00 for all Mini-Win unit numbers

n is 0-3, depending on plug address, and specifies a unit number

Some decoded addresses are not given as the addresses do not fit into the patterns outlined above.

The second byte of each 2-byte entry contains the status of the drive. Status codes and their meanings are defined as follows:

Code (hex)	Meaning
00	No disk in drive
01	Drive mounted with unlabelled volume and not in use
02	Drive mounted with labelled DOS volume and not in use
03	Drive mounted with labelled DOS volume and in use
04	Drive indisposed
05	Drive reserved
06	Drive open under direct control
07	Drive mounted with unlabelled volume and in use (Model 40 and EM-OIS only)

The number of 2-byte entries in the list is equal to the total number of drives configured in the system. This number can be determined through the Get System Configuration request. A maximum of 13 drives can be configured.

5.3.4 Get Drive Vector

The Get Drive Vector request is used to obtain specific information about a single disk drive. When the request is made, the 1-byte drive address of the desired disk drive is stored in the first byte (offset 0) of the data area. (Refer to Section 5.1.1.)

Request type: Immediate

Function code: x'06'
Data pointer: Pointer to data area containing drive address (upon request) and drive-specific information (upon completion)

If the request is unsuccessful, the first byte of the data area is set to x'FF' and the second byte is loaded with a status code (Model 40 and EM-OIS only). Status codes are defined as follows:

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, the master fills the data area with the following information:

Offset (hex)	Contents
01	Drive status code (same as that returned in second byte of Get Drive List data area, refer to Section 5.3.3). If the specified drive does not exist, an x'FF' is returned in this byte.
02	If the drive status is x'01' or x'07' (unlabelled volume mounted), this byte contains the previous mount code (refer to Section 5.3.3) If the drive status is x'04', x'05', or x'06', this byte contains the unit number of reserving slave.
03	If the previous mount code was x'87' (VAU map damaged) or if the disk is unlabelled and has suffered VAU map damage since the last mount, this byte contains the hardware error code from the volume label (Model 40 and EM-OIS only). Otherwise, see below.

If there is no DOS volume currently mounted on the specified drive, the remainder of the data area is filled with zeroes. Otherwise, the master fills the remainder of the area with the following information:

03-04	Number of file space VAUs
05-06	Number of free VAUs
07-08	Number of VAUs comprising catalog
09-0B	Date initialized
0C-13	Volume name
14	Password indicator: 00 Volume does not have a password 01 Volume has a password
15	Sectors per VAU
16-18	Date last mounted
19-1A	Catalog format level
1B	System disk flag: 00 Not a system disk 01 Is a system disk
1C-1E	Date last IPLed (if system disk)
1F-20	System release (if system disk)
21	Write protect code: 00 Not protected 01 Hardware protected 02 Catalog contains I/O error 03 Alternate VAU map contains I/O error 04 Primary VAU map contains I/O error
22	VAU map condition code: 00 Primary OK, alternate not allocated 01 Primary OK, alternate contains I/O error 02 Primary OK, alternate OK 03 Primary contains I/O error, alternate OK
23-3F	(Reserved)

5.3.5 Get Slave List

The Get Slave List request is used to obtain information about the slaves currently configured on the system. (Refer to Section 5.2.1.)

Request type: Immediate

Function code: x'0A'
Data pointer: Pointer to data area to receive list

If the request is unsuccessful, the first byte of the data area is set to x'FF' and the second byte is loaded with one of the following error codes (Model 40 and EM-OIS only):

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, the receiving area is filled with a list of 2-byte entries describing each slave currently configured on the system. The first byte of each entry contains the slave's unit number. The second byte contains the current status of the slave. The status codes and their meanings are defined as follows:

Code (hex)	Meaning
00	Slave is not IPLed
01	Slave is idle
02	Slave is running
03	Slave is under direct control without polling (Model 40 and EM-OIS only)
04	Slave is under direct control with polling (Model 40 and EM-OIS only)

The number of 2-byte entries in the list is equal to the total number of slaves configured in the system. This number can be determined with the Get System Configuration request. This number is always in the range of 0 to 31 (decimal).

5.3.6 Get Slave Vector

The Get Slave Vector request is used to obtain specific information about a slave. (Refer to Section 5.2.1.)

Request type: Immediate

Function code: x'0B'
Data pointer: Points to a 32-byte data area containing a unit number and unit-specific information.

When the request is made, the first byte of the data area contains the 1-byte unit number of the desired slave.

If the request is unsuccessful, the first byte of the receiving area is set to x'FF' and the second byte is loaded with one of the following error codes (Model 40 and EM-OIS only):

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, the master fills the data area with the following information:

Offset (hex)	Contents
01	Slave status code (refer to the list of status codes in Section 5.3.5). If the specified slave is not currently configured in the system, this byte contains an x'FF'.
02	Slave type code (see HM-04)
03	Slave activity code (refer to Section 9.11)
04	Slave's hardware status (see TM-01 and HM-04)
05	Last termination cause. The value in this byte indicates why the slave was last terminated. The defined codes are as follows: 00 Terminate request 01 Unrecoverable datalink transmission error 02 Memory parity errors 03 Powered off 04 Released from direct control (Model 40 and EM-OIS only) 05 XFS nontransmission error

06,07 Slave characteristics. These two bytes contain the values passed to the master with the Set Characteristics request and generally reflect the slave's local IN 7 and IN 8 switch settings.

08 Controlling unit number (Model 40 and EM-OIS only). If the current state of the slave is either 3 or 4 (under direct control), this byte contains the unit number of the controlling slave. Otherwise, undefined.

09-1F (Reserved)

5.3.7 Open DC

The Open DC request is used to open a disk for direct control. (Refer to Section 5.1.2.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'09'
02	Address of drive
04-05	Pointer to a namestring containing a DOS volume password and/or system name. If the value here is zero, no password of system name is specified.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex) 80 Operation successful 02 Addressed drive unavailable 03 No disk in drive 04 Namestring format error 06 Password missing or incorrect for requested access 07 Master processor overload 08 Permanent I/O error Model 40 and EM-OIS only: 09 Request or request option not supported 0D System not accessible 0E System connection broken 0F Network overload 12 XFS namestring format error
02	RN of disk.
03	Sectors per cylinder of disk (low-order byte).
06-08	Disk capacity (in sectors).

09 If status code in byte 1 equals x'08', this byte contains the hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)
If byte 1 equals x'02', this byte contains the current drive state. (Refer to Section 5.3.3 for a list of drive states.)

0A Sectors per cylinder of disk (high-order byte).

0B Disk Format:

0	256 bytes x 16 sectors
1	512 bytes x 9 sectors
02-FF	Reserved

5.3.8 Open DCS

(Model 40 and EM-OIS only)

The Open DCS request is used to open a slave for direct control. (Refer to Section 5.2.1.)

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'10'
02	Unit number of slave.
04-05	Pointer to a system namestring. If the values here is zero, no system name is specified.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation successful 03 Slave unavailable 04 Namestring format error 07 Master processor overload 08 Permanent I/O error 09 Request or request option not supported 0D System not accessible 0E System connection broken 0F Network overload 12 XFS namestring format error
02	RN of the slave.
09	If the status code in byte 1 equals x'08', this byte contains the hardware error code. (Refer to Section 3.2 for a list of hardware error codes.)

5.3.9 **Release Drive**

The Release Drive request is used to relinquish exclusive use of the specified disk drive. (Refer to Section 5.1.1.)

Request Type: Immediate

Function code: x'08'
Data pointer: Points to a 2-byte data area containing the address of the drive to be released and the completion status code.

If the request is unsuccessful due to errors in external handling, the first byte of the data area is set to x'FF' and the second byte is loaded with one of the following status codes (Model 40 and EM-OIS only):

Code (hex)	Meaning
07	Master processor overload
09	Request or request option not supported
0D	System not accessible
0E	System connection broken
0F	Network overload
16	Illegal release
17	XFS illegal release

If the request is successful, the second byte contains status code reflecting the outcome of the operation.

x'80' The release operation was successful.

x'FF' The specified drive does not exist in the system configuration.

Otherwise The specified drive was not previously reserved by the slave requesting the release. This byte contains the current state of the specified drive.

If the drive is no longer reserved or indisposed, the master performs a mount operation on the drive. In this case, the returned status byte contains a value x'8n', where "n" indicates the result of the attempted mount. The possible values of n are defined as follows:

n	Meaning
0	Mounted successfully
1	Mount failed -- Drive not ready
2	Mount failed -- Invalid or nonexistent volume label
3	Mount failed -- Duplicate volume name
4	Mount failed -- I/O error rewriting volume label
5	Mount failed -- Catalog damaged (first sector)
6	Mount failed -- Catalog damaged (free chain)
7	Mount failed -- VAU map damaged

5.3.10 **Reserve Drive**

The Reserve Drive request is used to request the exclusive use of a specified disk drive. (Refer to Section 5.1.1.)

Request type: Immediate

Function code: x'07'
Data pointer: Points to a 2-byte data area containing the address of the drive to be reserved and the completion status code.

If the request is unsuccessful due to errors in external handling, the first byte of the data area is set to x'FF' and the second byte is loaded with an error code. The error codes are defined as follows:

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, the second byte of the data area is loaded with a status code reflecting the outcome of the operation.

x'80' The drive is successfully reserved.

x'FF' The specified drive does not exist in the system configuration.

Otherwise If the drive cannot be reserved, this byte contains the current state of the specified drive (refer to Section 5.3.3).

If another drive connected to the same unit as the specifed drive cannot be marked "Indiposed" on behalf of the requesting slave, this byte contains the second drive's current state with the x'10' bit set.

NOTE

If the first byte of the data area does not contain a valid disk drive address, as listed in Section 5.3.3, the request will succeed but the reserve operation will not.

CHAPTER 6
EXTERNAL FILE SOURCES

In a standard OIS environment, the master processor supplies files and system information to slave processors. There are circumstances, however, when the master cannot supply this data directly but must turn to a second slave to obtain the data from another system. In this case, the second slave is known as an "external file source" or "external function source" (XFS) slave. A Wang Inter-System Exchange (WISE) box is an example of an XFS slave.

This chapter briefly describes the OIS external file source feature. Refer to the DOS External File Source Slave Interface Specification for complete details.

NOTE

This feature is available as a software option only on the Model 40 and EM-OIS.

6.1 ESTABLISHING AN XFS SLAVE

To establish itself as an external file source, a slave issues the Define File Source Connection request. The slave specifies the address of the master communication area (MCA) to be used by the master when issuing XFS slave requests. The MCA resides in the slave's memory and functions much like the slave/master communications area (SCA). It is 256 bytes long and can be anywhere in slave memory except page 0.

A slave disables itself as an XFS slave by reissuing the Define File Source Connection request with an MCA address of x'0000'.

6.1.1 System-Specific XFS Slaves

When issuing the Define File Source Connection request, the slave can optionally specify a system name. Doing so tells the master to pass the slave only those requests directed to the named system.

6.1.2 Watchdog Timer

Another option of the Define File Source Connection is the watchdog timer. The timer is activated by specifying the "activate timer" code in the request option byte and storing a nonzero value in the MCA.

When the watchdog timer is activated, the master periodically decrements the value in the MCA. If the slave fails to reset the value and the timer reaches zero, the master IPLs the slave. The slave's resources are returned to the system and any pending I/O requests directed through the slave are completed with an error status.

NOTE

The interval at which the master decrements this timer is determined by the slave polling rate; it should not be assumed to be any particular constant.

To deactivate the timer, the slave waits for all pending requests to complete (with the Quiesce request) and then reissues the Define File Source Connection request with the "no option" code in the option byte.

6.2 ACCESSING AN XFS SLAVE

A slave indicates that it wants to access the resources of a remote system (files, direct control devices, and so on) in one of two ways.

- By specifying the remote system name with the request
- By logically attaching (with the Attach request) to the remote system and then omitting a system name in any further requests

When the master realizes that it is not the intended system, it searches its list of defined XFS slaves to find a slave that can honor the request. If it does not find such a slave, the master returns a "system not accessible" status code to the requesting slave.

When the master finds an XFS slave that can process the request, it passes the request onto that slave (using the protocol defined in the DOS External File Source Slave Interface Specification). The original requesting slave is not polled again and no status code is returned until the request completes through the XFS slave. When the request does complete, the master passes the completion status code directly from the XFS slave to the requesting slave.

6.3 INTEGRAL REQUEST FIELD

When a request is being processed through an external file source slave, buffering limitations within the slave sometimes make it necessary to fragment a large I/O request into a series of smaller requests. However, fragmenting a request in this fashion can adversely affect the locking and unlocking of file segments at the remote system. To overcome this difficulty, an "integral request" field is included in the Read, Read Lock, and Write request control blocks.

When an external file source slave decides to fragment an I/O request, it sets the integral request field to some nonzero value. For Read and Read Lock requests, this value should be the total number of sectors to be read and/or locked; for Write requests it can be any nonzero value. When the last piece of the fragmented request is sent, the XFS slave sets the integral request field to zero, unless the field was nonzero when the request was originally received. In that case, the original request was a fragment of a larger request from another XFS slave; the field should remain nonzero, even if the slave does not fragment it further.

The original requesting slave should always set this field to zero.

6.4 REQUEST FORMATS

6.4.1 Define File Source Connection

(Model 40 and EM-OIS only)

The Define File Source Connection request is used by a slave to enable or disable itself as an external file source (XFS) slave -- a slave that supplies files and system information to the master. (Refer to Section 6.1.)

Request type: Immediate

Function code: x'12'
Data pointer: Points to a 7-byte control block

The requesting slave initializes the control block with the following information:

Offset (hex)	Definition
02-03	MCA address. If this value is x'0000', the slave is disabling itself as an XFS slave.
04-05	Pointer to an optional delimited system name (in the form /SYSTNAME;/, where the semicolon is required)
06	Option byte: 00 No option 01 Activate watchdog timer 02 Terminate MCA requested (System name pointer must be set to x'0000') 03-FF (Reserved -- treated as "activate timer")

The master completes the control block as follows:

Offset (hex)	Definition
00	Primary status (in hex) 80 Request successful FF Request unsuccessful
01	Secondary status (valid when primary status = x'FF') 04 Namestring format error 09 XFS not supported/invalid request 18 MCA currently in use 19 System name table full

CHAPTER 7
STARTUP AND TERMINATION

Many different software subsystems can run under the OIS disk operating system. In order to permit simultaneous use of different subsystems, slave startup is handled in a very general way.

7.1 PROGRAM FILES

In order to run a program in a slave processor, the user must prepare a disk file containing a memory image of the program. The program can use memory pages 1 upward. Each sector of the file corresponds to one page of memory (sector 0 with page 1, sector 1 with page 2, and so on), with the length of the file equal to the size of the program. The program entry point must be at location x'0100'.

7.2 SLAVE SERVICES

The first page of each slave's memory contains a small "supervisor" consisting of two slave service routines. These routines are invoked with the RST 6*8 and RST 7*8 instructions and require the SCA function area to be located at address 4 of slave memory (refer to Section 2.1). A program can make use of these services only if it does not destroy the contents of memory locations x'30' through x'BF'.

7.2.1 Starting a File

This service routine (invoked by the RST 7*8 instruction) is used to terminate program processing by passing control to another program residing in a program file. There are two methods of invoking this routine. The first method explicitly names the file to be loaded, using a programming sequence similar to the following:

```
        LD    HL,NAME
        XOR   A
        RST   7*8
         .
         .   (error handling)
         .
NAME DB  "/name-of-file/"
```

The second method specifies the reference number of a previously opened file, using a programming sequence similar to the following:

```
LD    HL,x'0000'
LD    A,RN          ;this is the file reference #
RST   7*8
```

The specified file is read into memory, starting at page 1; the remainder of memory is left unchanged. The file is closed after being read. Control is then passed to location x'100' with interrupts disabled. The BC and DE processor registers are left unchanged; the PSW and HL registers contain garbage; and the SP register contains x'100' (Refer to Section 2.5).

If any errors are encountered while opening an explicitly named file, control is transferred to the instruction immediately following the RST; the Open error code is in the A register.

If any errors are encountered while reading or closing the file, or if the A and HL registers are not properly loaded, the service routine terminates to the master, resulting in the slave being IPLed.

7.2.2 Voluntary Termination

This service routine (invoked by the single instruction, RST 6*8) is used to terminate program processing. This method of program termination is equivalent to issuing the Terminate master request (refer to Section 7.4.3).

7.3 SLAVE IPL AND RE-IPL

7.3.1 Auto-IPL Slaves

Unless specified otherwise when the system is configured (refer to Section 7.3.2), all slaves are IPLed with a startup program under the following circumstances:

- At the conclusion of the master IPL procedure
- Whenever the slave issues a Terminate request
- Whenever the master detects that the slave has just powered on or has gone into the "parity error" state
- Whenever a slave is opened for direct control (Model 40 and EM-OIS only)
- Whenever a direct control slave is closed by its controlling slave and is not being polled by the operating system (Model 40 and EM-OIS only)

When a slave is IPLed, all its open files are closed and all other system resources it is using are returned. If the slave is being opened for direct control, program DOS.DCSTART is run to clear the slave's memory to zero. This is done as a security measure to prevent the controlling slave from having access to potentially sensitive data in the slave's memory. (Refer to Section 5.2.)

If the slave is not being opened for direct control, program DOS.START is run. If DOS.START cannot be opened, the slave is placed in a dead loop with its activity code set to "not operational".

If DOS.START is loaded successfully, it attempts to start program DOS.STARTn, where "n" is the slave type code. If DOS.STARTn is not found and the device is not a workstation, program DOS.BATCH is loaded and the batch handler program is run. If DOS.BATCH is not found, the slave is placed in a dead loop with its activity code set to "not operational".

If DOS.STARTn is not found but the device is a workstation, program DOS.MENU is loaded and the system main menu program is run. If DOS.MENU is not found, the slave is placed in a dead loop with its activity code set to "not operational".

7.3.2 Manual-IPL Slaves

When the system is configured, a number of slaves can be specified as manual-IPL slaves. (The number specified is always allocated from those slaves with the highest unit numbers.) Manual-IPL slaves are loaded in the same fashion as auto-IPL slaves, with the following exceptions:

- If a manual-IPL slave enters a parity state or develops an unrecoverable datalink error, it is not IPLed automatically.

- When DOS.START is loaded into a manual-IPL slave, it attempts to start DOS.STARTuu, where "uu" is the unit number of the slave. If that file does not exist, the slave is loaded in the same way as an auto-IPL slave.

7.3.3 Date and Time During IPL

Since date and time are necessary for maintenance of the system catalog as well as various other applications, the current date and time must be entered each time the operating system is IPLed.

When the MASTER RESET button is pressed, the system initialization procedure sets the current date and time to zeroes. Thus, any slave requesting the date and time can tell whether the date has been set by checking the month for a value of zero. If the month is zero, the system is in the "date not set" state. When a slave is IPLed with the system in this state, program DOS.IPL is run rather than the normal startup program.

When running in a workstation, DOS.IPL asks the operator to enter the current date and time and to press EXECUTE after verifying the entry. When the program has a complete entry, it issues a Set Time and Date request and then re-IPLs the slave. In other slave processors, DOS.IPL continuously monitors the current date and time. When it detects that the date has been set, it immediately re-IPLs the slave.

7.4 REQUEST FORMATS

7.4.1 Get Time and Date

The Get Time and Date request is used to obtain the current time and date. (Refer to Section 7.3.3.)

Request type: Immediate

Function code: x'02'
Data pointer: Points to 6-byte data area to receive time and date information

The 6-byte data area is filled in by the master in the following format:

Byte	Contents (decimal)
0	Second (0-59)
1	Minute (0-59)
2	Hour (0-23)
3	Day (1-31)
4	Month (1-12)
5	Year (0-99) [+ base of 1900]

NOTE

This 6-byte data area shares a common format with the Set Time and Date request.

7.4.2 Set Time and Date

The Set Time and Date request is used to specify the current time and date for the system. (Refer to Section 7.3.3.)

Request type: Immediate

Function code: x'09'
Data pointer: Points to a 6-byte data area containing new date and time information.

The 6-byte data area contains the date and time in the following format:

Byte	Contents (decimal)
0	Second (0-59)
1	Minute (0-59)
2	Hour (0-23)
3	Day (1-31)
4	Month (1-12)
5	Year (0-99) [+ 1900]

NOTE

This 6-byte data area has the same format as that of the Get Time and Date request.

7.4.3 **Terminate**

The Terminate function is used to terminate processing of a program or subsystem. (Refer to Section 7.2.2.)

Request type: Immediate

Function code: x'03'
Data pointer: Not used

All open files, disks open for direct control, slaves open for direct control, reserved drives and other system resources controlled by the slave are closed or returned, and the slave is reloaded with the appropriate startup program.

CHAPTER 8
LOCAL I/O OPERATIONS

The OIS disk operating system supports slave access of the OIS file structure on disks mounted in the slave's local archive drive. This chapter discusses how to use this local I/O package.

8.1 LOADING THE LOCAL I/O PACKAGE

The local I/O package (LIO) runs in the upper portion of memory in the 64K local archiving workstation. Before it can be accessed, however, the slave program must load it into memory.

LIO is stored on the system disk in program file SYSTEM.LIO. Sector 0 of this file contains the information necessary to load the remainder of the file into memory. Sector 0 has the following format:

Byte (hex)	Contents
00	LIO load address (stored as a page number).
02	File-relative sector number (FRSN) of the beginning of LIO within SYSTEM.LIO.
03	Length of LIO, in sectors.

Using standard system file I/O requests (refer to Chapter 4), the slave program must open SYSTEM.LIO and load the necessary sectors into the workstation memory at the specified load address. SYSTEM.LIO must remain open until LIO terminates.

8.2 ACCESSING THE LOCAL I/O PACKAGE

8.2.1 Jump Table

The local I/O package is entered via a jump table located at the LIO load address, as specified in sector 0 of SYSTEM.LIO. The jump table consists of one 3-byte entry for each LIO routine and has the following format:

Offset (hex)	Contents
00	Initialization routine entry point
03	Resume routine entry point
06	Suspend routine entry point
09	Terminate routine entry point
0B	Request handler entry point

8.2.2 User Variable Area

On entry, each LIO routine expects a pointer to the user variable area in register BC.

The user variable area is 16 bytes long and can be located anywhere in slave memory. It contains information about the current state of the LIO package. Some of this information is supplied by the slave program and some is supplied by LIO, itself. The format of this area is as follows:

Byte (hex)	Contents
00	File reference number (RN) of SYSTEM.LIO. This value is obtained by the slave program when it opens SYSTEM.LIO.
01	LIO page address (also the page address of the jump table). This value is obtained by the slave program from byte 0 of sector 0 of SYSTEM.LIO.
02-03	File-relative sector number (FRSN) of the beginning of LIO within file SYSTEM.LIO. This value is obtained by the slave program from bytes 1 and 2 of sector 0 of SYSTEM.LIO.
04	Length of LIO, in sectors. This value is obtained by the slave program from byte 3 of sector 0 of SYSTEM.LIO.
05	File reference number (RN) of scratch file used to hold control and event handler information when LIO is temporarily suspended. This value is obtained by LIO when it opens the scratch file.
06	In-memory flag (set by LIO): 0 = not in memory 1 = in memory
07	Current state of the local drive (refer to Section 5.3.3 for a list of possible drive states). This value is set by LIO.
08-0F	Reserved

8.2.3 Status Codes

The local I/O package returns a status code to the user (in register A) following any request through the jump table. This status code indicates only whether LIO accepted the request. It does not indicate the status of the requested operation. The status codes are defined as follows:

Code (hex)	Meaning
80	Request accepted
01	Unable to load overlay
02	LIO unavailable for this slave
03	Unable to save control blocks and event handler on system disk
04	Unable to read control blocks and event handler file
05	Insufficient space for TCB
06	Insufficient space for VCB
07	Insufficient space for FCB
08	Insufficient space for VAU map buffer
09	Insufficient space for catalog buffer
0A-FF	Reserved for future use

8.3 INITIALIZING THE LOCAL I/O PACKAGE

To initialize the local I/O package, the slave program must first load the package and then invoke the initialization routine via the jump table.

The initialization routine first determines whether the workstation has a local drive attached. If it does not, LIO returns the "unavailable for this slave" status code; a slave obviously cannot use the local I/O package unless it has a local drive.

Once the initialization routine establishes that the slave has a local drive, it does the following:

- Sets up either the regular or the mini local event handler. Only one event handler can be present in memory.

- Sets the flag in the user variable area to indicate that LIO is currently in memory.

- Mounts the local drive and posts the status in the user variable area.

- Sets the number of pages used by the slave program (byte X'0011' of slave memory, refer to Section 2.2) to account for LIO.

8.4 LOCAL I/O REQUEST HANDLER

8.4.1 Request Processing

Once the local I/O package is active, all requests should be channeled through the LIO request handler, regardless of whether or not the request is directed at the local drive. The request handler determines which requests are directed to the local drive and which requests are to be passed to the OIS master.

Most immediate requests are passed directly to the master, while most of the nonimmediate requests can be directed to the local drive. The slave program indicates that the local drive is required by specifying the local volume name in the approrpiate namestring or by specifying a drive address of x'00'.

Requests are channeled to the LIO request handler by putting a data pointer into the slave/master communication area (SCA) of workstation memory (locations x'0005' and x'0006', refer to Section 2.1), the function code into the E register, and a pointer to the user variable area in register BC. Control is then transferred to the request handler via a jump to offset x'0B' of the jump table.

If the request handler determines that the request should be passed to the OIS master, it loads the request code into the SCA function code (byte x'0004' of the workstation memory) to be retrieved by the master on its next poll. The request handler then returns control to the slave program; the slave program is responsible for waiting for the request to complete. If the request handler determines that the request is for the local drive, it processes the request and posts the status of the operation before returning control to the slave program.

The local I/O package can and will disable interrupts while processing. Thus, there must be no data-link block transfers executed while the local drives are running. LIO waits for all master operations to complete (by issuing the Quiesce request) before processing a local request. Note, therefore, that the slave program should not attempt to overlap master requests with local drive operations.

While accessing a local disk drive, the local I/O package changes the Z80 interrupt mode. However, the mode cannot be restored to the slave program's previous mode since the Z80 does not allow the software to determine this value. Therefore, LIO automatically sets the processor to mode 0 upon completion of a local request.

The following subsections show how LIO handles immediate and nonimmediate requests.

Immediate Requests

Code	Request	Directly to OIS Master?
00	Idle Drive	No; if drive = x'00' then LIO idles local drive
02	Get Time and Date	Yes
03	Terminate	No; LIO must perform clean-up
04	Get System Configuration	No; LIO must adjust the number of drives
05	Get Drive List	No; LIO must add local drive to list
06	Get Drive Vector	No; if drive = x'00' LIO gets local drive information
07	Reserve Drive	No; if drive = x'00' LIO reserves the local drive
08	Release Drive	No; if drive = x'00' LIO releases the local drive
09	Set Time and Date	Yes
0A	Get Slave List	Yes
0B	Get Slave Vector	Yes
0C	Set Slave Activity	Yes
0D	Suppress Polling	Yes
0E	Set Signal List	Yes
0F	Query Signal	Yes
10	Send Signal	Yes
11	Quiesce	Yes
12	Define File Source Connection	Yes
13	Attach	Yes
14	Set Characteristics	Yes
15	Get System Names	Yes
16	Move Function Area	Yes

Nonimmediate Requests

Code	Request	Criteria for local request
00	Open	Local volume name
01	Read	Valid local I/O RN
02	Write	Valid local I/O RN
03	Catalog	Valid local I/O RN
04	Close	Valid local I/O RN
05	Opencat	Local volume name
06	Read Lock	Valid local I/O RN

07	Set EOF	Valid local I/O RN
08	Reopen	Valid local I/O RN
09	Open DC	Drive address = x'00'
0A	Direct IO	Valid local I/O RN
0B	Lock	Valid local I/O RN
0C	Get File Name	Local volume name
0D	File Lock	Valid local I/O RN
0E	File Unlock	Valid local I/O RN
0F	Assign Password	Local volume name
10	Open DCS	(Never for local I/O)
11	Supervisory Read	(Never for local I/O)
12	Get System List	(Never for local I/O)

8.4.2 **Restrictions**

Number of Open Files

Due to space constraints, the local I/O package can support a maximum of five open files on the local disk at one time. If the slave program attempts to open more than five files, a "master processor overload" error is returned.

During the Opencat request, this maximum is reduced to three open files on the local disk.

Slave Termination

When the local I/O package is active, slave programs cannot invoke the slave service terminate routine (via RST 6*8) directly. The Terminate request must be used (and passed through the LIO request handler) to insure that all files on the local disk have been closed.

Use of the Move Function Area Request

The local I/O package requires the SCA function area to begin at location x'04' of the workstation memory (refer to Section 2.1). Therefore, the slave program must not use the Move Function Area request while the local I/O package is active.

Memory Mode Selection

Since the local I/O package resides in the upper portion of the workstation's memory, the slave program must ensure that memory mode is selected before calling LIO. Selecting screen mode will result in unpredictable results.

8.4.3 Idling the Local Drive

Since local drives have a limited life expectancy, it is important to put the local drive in the 'idle' state when it is not being accessed. The local I/O package automatically puts the drive in the idle state after processing certain local requests, as well as any time an error condition is detected. The local drive is automatically idled after the following operations:

- Assign Password
- Close
- Get File Name
- Opencat
- Release Drive
- Reserve Drive
- Terminate

The slave program can also explicitly request that the drive be idled by issuing the Idle Drive request. This request causes LIO to turn off the drive motor and lift the heads. It has no meaning when directed to the OIS master.

8.5 SUSPENDING THE LOCAL I/O PACKAGE

If necessary, the local I/O package can be suspended and temporarily removed from memory. This permits programs that are not accessing the local drive to have full use of the workstation memory. LIO is suspended by invoking the suspend routine via the jump table.

The suspend routine does the following:

- Issues an Idle Drive request for the local drive.

- Resets the number of pages used by the slave program (byte x'0011' of slave memory) to its previous value.

- Clears the in-memory flag in the user variable area to indicate that LIO is not in memory.

- Opens a scratch file on the OIS system disk and writes out the LIO control blocks and event handler. The reference number of this file is saved in the user variable area.

Note that files on the local disk are not closed automatically when the local I/O package is suspended.

8.6 RESUMING THE LOCAL I/O PACKAGE

When the local I/O package has been suspended, it can be resumed by invoking the resume routine via the jump table. The slave program must first reload LIO from the program file SYSTEM.LIO, as described in Section 8.1.

The resume routine uses the scratch file reference number (previously saved in the user variable area) to read the control blocks and event handler. It then closes the scratch file, sets the in-memory flag to indicate that LIO is resident, and adjusts the number of pages in use by the slave program (byte X'0011' of slave memory) to account for the size of the LIO package.

8.7 TERMINATING THE LOCAL I/O PACKAGE

When all local I/O activity is done and the slave program has no further need of the local I/O package, LIO should be terminated. This is done by invoking the terminate routine via the jump table. (Note that invoking the terminate routine is meaningful only if LIO has been initialized.)

The terminate routine does the following:

- Closes all files opened on the local disk.
- Idles the local drive.
- Unlocks the door of the local drive.

The slave program is responsible for closing SYSTEM.LIO on the system disk.

8.8 REQUEST FORMATS

8.8.1 Idle Drive

The Idle Drive request is used to place a disk drive in the idle state. (Refer to Section 8.4.3.)

Request type: Immediate

Function code: x'00'
Data pointer: Points to a 2-byte data area containing the address of the drive to be idled and a completion status code.

The first byte of the data area must contain a valid disk drive address. The local drive always has an address of x'00'.

The Idle Drive request never fails; a successful completion status code of x'80' is always returned in the second byte of the data area.

CHAPTER 9
MISCELLANEOUS SYSTEM REQUESTS

This chapter describes a set of miscellaneous requests that a slave can use to obtain information and perform general functions.

9.1 Attach

(Model 40 and EM-OIS only)

The Attach request is used to logically connect a slave to another master system. (Refer to Section 3.3.)

Request type: Immediate

Function code: x'13'
Data pointer: Points to a 7-byte control block

The requesting slave initializes the control block with the following information:

Offset (hex)	Definition
04-05	Pointer to a delimited system name (in the form /SYSTNAME;/, where the semicolon is required)

The master completes the control block as follows:

Offset (hex)	Definition
00	Primary status (in hex) 80 Request successful FF Request unsuccessful
01	Secondary status (valid when primary status = x'FF') 04 Namestring format error 09 Request or request option not supported

Other bytes in the control block are reserved.

Note that the successful completion of this request does not mean that the specified system exists.

9.2 Get System Configuration

The Get System Configuration request is used to obtain information about the current system.

Request type: Immediate

Function code: x'04'
Data pointer: Points to a 32-byte data area to receive system configuration information

If the request is unsuccessful, the first byte of the receiving area is set to x'FF' and the second byte is loaded with an error code. The error codes are defined as follows:

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, the receiving area is filled in by the master with the following information:

Offset (hex)	Contents
00	System release number
01	System level number
02	Maximum number of slaves configured in the system
03	Maximum number of drives configured in the system
04	Drive address of system residence volume
05-0C	Volume name of system residence volume
0D-14	Program Option Masks

Offset (hex)	Bit	Contents (Model 40 and EM-OIS)
0D	7	(Reserved -- always 1)
	6	(Reserved -- value undefined)
	5-4	Second master switch bank bits 6-5 (switches 7-6)
	3	Second master switch bank bit 4 (switch 5) This is the OIS-W (Wangnet) option bit.

		2	Second master switch bank bit 3 (switch 4) This is the OIS-200 (Alliance) option bit.
		1	Second master switch bank bit 2 (switch 3) This is the BASIC compiler option bit.
		0	(Reserved -- always 0)
	0E	7	Second master switch bank bit 1 (switch 2) This is the C program language option bit.
		6	Second master switch bank bit 0 (switch 1) This is the 125A option bit.
		5	(Reserved -- always 0)
		4-0	First master switch bank bits 6-2 (switches 5-1) Note: The OIS-40, -50, -60, and -70 do not have this switch bank and return zeroes.
	0F	7	Always 1
		6-3	(Reserved -- always 1s)
		2	If 1, EM is not configured If 0, EM is configured
		1	(Reserved -- always 1)
		0	If 1, XFS is not configured If 0, XFS is configured
	10-14	7-0	(Reserved -- always 1s)
15-16			Configured number of resource units (Model 40 and EM-OIS only)
17-18			Number of resource units in use (Model 40 and EM-OIS only)
19			Unit number of lowest manual IPL slave (Model 40 and EM-OIS only)
1A-1F			Software configuration switches (Model 40 and EM-OIS only)

The number of slaves and disk drives are used in conjunction with the Get Drive List and Get Slave List requests.

9.3 Get System List

(Model 40 and EM-OIS only)

The Get System List request is used to obtain a list of the current systems to which a slave can attach.

Request type: Nonimmediate

The requesting slave provides the following information in the indicated fields of the request control block:

Byte (hex)	Contents
00	Request Code = x'12'
03	Maximum number of system name entries to be returned.
04-05	Pointer to the area to receive the data. Each entry requires 16 bytes.

The master returns the following information in the indicated fields of the request control block:

Byte (hex)	Contents
01	Status code (in hex): 80 Operation successful 09 Request or request option not supported
04-05	The data area pointed to by the value in this location is filled with 16-byte system name entries. The local system name is always the first entry. The format of each entry is as follows:

Byte (hex)	Contents
0-7	System name, left-justified, padded with zeroes
8-F	Reserved, currently all zero

Byte (hex)	Contents
09	Number of entries returned.

NOTE

Because of the dynamic nature of external file sources, this request can only provide a "best guess" list of current system names.

9.4 Get System Names

(Model 40 and EM-OIS only)

The Get System Names request is used by a slave to determine the names of the systems to which it is currently attached physically and logically (via an Attach request).

Request type: Immediate

Function code: x'15'

Data pointer: Points to a 16-byte data area to receive the physical and logical system names of the requesting slave

The receiving area is filled by the master with the following information:

Offset (hex)	Contents
00-07	Physical system name of the requesting unit.
08-0F	Logical (or current) system name of the requesting unit. The current system is the one to which the slave is attached.

9.5 Move Function Area

(Model 40 and EM-OIS only)

The Move Function Area request moves the SCA function area to the indicated address in slave memory.

Request type: Immediate

Function code: x'16'
Data pointer: Points to a 3-byte data area to be used for the SCA Function Area.

For the sake of convenience, the format of the function area is repeated here. Refer to Section 2.1 for a complete description.

Offset (hex)	Contents
00	Function code. This byte should be initialized to zero by the slave before the request is made.
01-02	Control block/data pointer (low-order byte first)

WARNING

Use this request with caution. Once the request has been issued, there is no way to interrogate the master about the new location of the function area.

9.6 Query Signal

The Query Signal request is used to interrogate a slave's own signal list. A Set Signal List request must have been issued previously in order for this request to be meaningful.

Request type: Immediate

Function code: x'0F'
Data pointer: Points to a 2-byte data area to receive the query status

The master searches the slave's signal list for a nonzero counter. If one is found, it is loaded into the first byte of the 2-byte data area and then reset to zero. The counter indicates the number of times the same signal was sent to the requesting slave. The second byte of the data area is loaded with the offset (into the slave's signal list) of the corresponding signal code.

If no nonzero counters are found, the first byte of the 2-byte data area is set to zero.

9.7 Quiesce

The Quiesce request is used to wait for all pending nonimmediate requests to complete.

Request type: Immediate

Function code: x'11'
Data pointer: (Not used)

When the master acknowleges the request by loading a x'00' in the SCA function code, the slave can assume that all pending nonimmediate requests (for that slave) have been completed and acknowledged.

9.8 Send Signal

The Send Signal request can be used by any slave to signal other slaves.

Request type: Immediate

Function code: x'10'
Data pointer: Points to a 3-byte data area containing the signal code being sent and a completion status code

The first two bytes of the data area contain the signal code.

If the request is unsuccessful, the first byte of the data area is set to x'FF' and the second byte is loaded with one of the following status codes (Model 40 and EM-OIS only):

Code (hex)	Meaning
07	Master processor overload
0D	System not accessible
0E	System connection broken
0F	Network overload

If the request is successful, every slave with the specified signal code defined in its signal code list (via a Set Signal List request) has its corresponding counter incremented by one. The number of slaves that are signaled is returned to the requesting slave in the third byte of the data area.

9.9 Set Characteristics

The Set Characteristics request is used by a slave to tell the master its characteristics.

Request type: Immediate

Function code: x'14'
Data pointer: Points to a 2-byte data area containing slave characteristics information.

The two bytes provided are recorded by the master and then listed in the slave vector of the requesting slave (refer to Section 5.3.6 for a discussion of the slave vector). The master assigns no further meaning to these values.

The two bytes provided are generally the values of the local IN 7 and IN 8 switch settings. These bytes are cleared when the slave is IPLed.

9.10 Set Signal List

The Set Signal List request is used to inform the master that the requesting slave can be signaled by other slaves.

Request type: Immediate

Function code: x'0E'
Data pointer: Points to a list of 2-byte signal codes

Signals are accepted for only the specified list of codes. Up to four 2-byte entries are allowed. Any entries beyond the first four are ignored.

The slave can also request that signaling be disabled by providing an empty list (that is, a list starting with two bytes of zero).

The master stores the signal codes set by each slave and maintains a 1-byte counter for each code set. When the Set Signal List request is issued, all counters are set to zero. Signals can be sent by any slave, using the Send Signal request. (The Send Signal request accepts one 2-byte signal code.) When a signal is sent, the master scans the signal lists of all slaves, testing for that signal. When it finds a slave with that signal code set in its list, the corresponding counter is incremented by one.

Later, when a slave issues a Query Signal request, the first nonzero counter is returned to the slave (along with the offset into the signal list) and the counter is reset to zero. If no nonzero counter is found, a zero status is returned to the slave.

The batch handler slave programs are a good example of the use of signals. All batch handlers are responsible for maintaining three queues and use the arrival of a signal to indicate that an item has been enqueued. The conventional mapping from queue name to signal is as follows:

```
QUEUE.Unn maps to       signal number x'00nn'
QUEUE.Tnnn    maps to       signal number x'0nnn'
QUEUE.Cn  maps to       signal number x'100n'
```

By convention, other signals are reserved for specific applications. Signals reserved thus far include:

```
x'A0nn'       for TC
x'F0nn'       for DVX
```

To avoid confusion and irratic system results, signals numbers should be unique to the application and environment.

9.11 Set Slave Activity Code

The Set Slave Activity Code request permits a slave to provide the master a 1-byte code indicating what the slave is doing. This code is then returned with any Get Slave Vector request specifying the slave.

Request type: Immediate

Function code: x'0C'
Data pointer: Points to a 1-byte activity code

The activity code is maintained by the master until one of the following events occurs:

- Another Set Slave Activity Code request is issued by the slave.
- The slave is terminated or re-IPLed.
- The system menu program is loaded (for workstation slaves).

NOTE

By convention, whenever a slave is idle, its activity code is set to x'00'.

A partial list of activity code assignments follows:

Code (hex)	Activity
00	Idle
01	Control functions
02	Program development
03	System utilities
04	Demonstrations
05	User-invoked program (other)
06	System generation
07	Logged on and idle
08	Supervisory Functions
70	Telecommunciations (TC) menus
71-77	TC normal operation
78	TC initialization
79	TC failure
7A-7F	TC reserved

80	BASIC
81	Word Processing
82	LISP
83	PLM80
84	PASCAL
85	Font Development
86	FORTRAN80
87	Diagnostic Monitor
88	DataBase
89	Ideographic Word Processing
8A	"C"
8B	CP/M
8C	Teletex
8D	Charter
90	Visual memory
91	Message system
92	Notebook
93	Calendar
94	Directory
95	The Data Manager
96-9D	Reserved
9E	UNIX
9F	WP Plus
A0	Printing
A1	Typesetting
A2	Scanning
A3	Warming up
A4	Output bin full
A5	Add toner
A6	Processing
A7-A8	(Reserved)
A9	Paper jam
AA	Change paper
AB	Change font
AC	Change ribbon
AD	Cover open
AE	Deselected
AF	Malfunction
B0	Voice processing
B1	Transcribing/Word Processing
B2	Voice (B2)
B3	Voice (B3)
B4	DVX Diagnostics
B5	DVX SOP Active
B6	DVX SOP backup
B7	DVX Incoming call
B8	DVX Calling user
b9	DVX Calling non-user
BA	DVX Idle
BB	DVX Offline
BC	DVX Reloading
BD	DVX Initializing
BE-BF	Reserved

D0	Creating index
D1	Backing up
D2	Restoring
D3	Waiting for error read
D4	Resubmitting errors
D5	Waiting to mount tape
D6	Waiting for next tape
D7	Being persued
FC	Program sequencing error
FD	Diagnostic failure
FE	Not operational
FF	Nonspecific busy

9.12 SUPERVISORY READ
(Model 40 and EM-OIS only)

The Supervisory Read request is by the OIS operating system group for debugging and testing. It allows system security to be circumvented. It should not be used outside of this group.

Items Supplied:

Byte (hex)	Contents
00	Request code - x'11'
02	Source identifier (in hex): 00 Master I-space 01-FD Slave unit number For the special performance master: 01-FC Slave Unit Number FD Clear Performance Data Buffer FE Master Performance Data FF Master D-space
04-05	Pointer to delimited password (not needed if performance data is requested)
06-07	Source address in master or slave (not needed if performance data is requested)
08-09	Destination address
0A-0B	Number of bytes to transfer (not needed if performance data is requested)

-Note: An incorrect password causes IPL of the requesting slave!-

Items Returned:

Byte (hex)	Contents
01	Status code (in hex) 80 Successful super read
03	Slave not available 08 Permanent I/O error 09 Request or request option not supported
09	Hardware error code if status = x'08'.

(Continued on Next Page)

If performance data of the regular master is requested, the Master returns 256 bytes of information at the destination address. This receiving area is filled with the following information (all items are in hexadecimal):

Bytes (hex)	Contents
00-07	Current date and time: 00 Thousandth of a second 01 Tenth of a second 02 Second 03 Minute 04 Hour 05 Day 06 Month 07 Year
08-09	Number of non-XFS immediate requests
0A-0B	Number of non-XFS non-immediate requests
0C-0D	Number of non-XFS read and read lock RCBs
0E-0F	Number of non-XFS write RCBs
10-11	Number of non-XFS sectors read (for above read and read lock RCBs)
12-13	Number of non-XFS sectors written (for above write RCBs)
14-15	Number of non-XFS IOQEs enqueued
16-17	Number of one-byte read transmission errors
18-19	Number of one-byte write transmission errors
1A-FF	Reserved

All counters are initialized to zero at system IPL. On overflow, they wrap back to zero.

9.13 Suppress Slave Polling

The Suppress Slave Polling request is used by a slave when it wants the master to stop polling it for a period of time. This function is often needed by slaves that must drive local DMA devices.

Request type: Immediate

Function code: x'0D'
Data pointer: Points to one byte containing a count of the number of tenths of a second for which polling is to be stopped

The slave provides one byte containing the length of time, in tenths of a second, for which polling should be stopped. This time period begins when the master acknowledges the request (by setting the SCA function code to zero).

Note that this function only affects the polling operation; it does not affect outstanding file requests. If the slave wants to stop all master/slave memory referencing, it must wait for all pending requests to complete (using the Quiesce request) before issuing the Suppress Slave Polling request.

CHAPTER 10
NONIMMEDIATE REQUEST STATUS CODES

The following tables provide a cross-reference of nonimmediate requests and their error codes.

Request	Acronym	Full Request Name
x'00'	OPEN	Open
x'01'	READ	Read
x'02'	WRITE	Write
x'03'	CAT	Catalog
x'04'	CLOSE	Close
x'05'	OCAT	OpenCat
x'06'	RLOCK	Read Lock
x'07'	SEOF	Set EOF
x'08'	ROPEN	Reopen
x'09'	ODC	Open DC
x'0A'	DIO	Direct I/O
x'0B'	LOCK	Lock
x'0C'	GFN	Get File Name
x'0D'	FLOCK	File Lock
x'0E'	FUNLK	File Unlock
x'0F'	APW	Assign Password
x'10'	ODCS	Open DCS
x'11'	SREAD	Supervisory Read
x'12'	GSYSL	Get System List

Request code

Error Code	Abbreviated Message Text	ORWCCORSRODLGFFA OSG PERALCLEODIOFLUPDRS EAITOAOOPCOCNONWCES NDTSTCFE KCLCSAS ETEKN KKDL	0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12
1	RN not Valid	! !X!X!X!X! !X!X!X! !X!X! !X!X! ! ! !	
	File name not found on volume	!X! ! ! ! ! ! ! ! ! ! ! !X! ! ! ! ! !	
	Node not found	! ! ! ! ! !X! ! ! ! ! ! ! ! ! !X! ! ! !	
2	Access conflict	!X! ! ! ! ! ! ! !X! ! ! ! ! ! ! ! ! ! !	
	Buffer not on page boundary	! !X!X! ! ! !X! ! ! ! ! ! ! ! ! ! ! !	
	New name already exists	! ! ! !X! ! ! ! ! ! ! ! ! ! ! ! ! ! !	
	Name indicates file	! ! ! ! ! !X! ! ! ! ! ! ! ! ! ! ! ! !	
	Drive not available	! ! ! ! ! ! ! ! ! !X! ! ! ! ! ! ! ! !	
	End of catalog reached	! ! ! ! ! ! ! ! ! ! ! ! !X! ! ! ! ! !	
3	Volume not found	!X! ! ! ! !X! ! ! ! ! ! !X! ! !X! ! !	
	No write access	! ! !X!X! ! ! !X! ! ! ! ! ! ! ! ! ! ! !	
	File not open R/W shared	! ! ! ! ! ! !X! ! ! ! !X! !X!X! ! ! !	
	No disk in drive	! ! ! ! ! ! ! ! ! !X! ! ! ! ! ! ! ! !	
	Slave/Disk not open DC	! ! ! ! ! ! ! ! ! ! !X! ! ! ! ! ! ! !	
	Slave not available	! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! X! X!	
4	Namestring format error	!X! ! !X! !X! ! ! !X! ! !X! ! !X! X! !	
	Segment out of bounds	! !X!X! ! ! !X! ! ! ! ! ! ! ! ! ! ! !	
	No file space available	! ! ! ! ! ! ! !X! ! ! ! ! ! ! ! ! ! !	
5	Tree structure error	!X! ! !X! !X! ! ! ! ! ! !X! ! !X! ! !	
	Lockout	! !X!X! ! ! !X!X! ! ! !X! !X! ! ! ! !	
6	Password missing or incorrect	!X! ! !X! ! ! ! ! !X! ! ! ! ! !X! ! !	
	Segment not locked	! ! !X! ! ! ! !X! ! ! ! ! ! ! ! ! ! !	
	Buffer not on page boundary	! ! ! ! ! !X! ! ! ! ! ! ! ! ! ! ! ! !	
	Password insuf. for new access	! ! ! ! ! ! ! ! !X! ! ! ! ! ! ! ! ! !	

```
!
        File not locked ! ! ! ! ! ! ! ! ! ! ! ! ! !X! !  !  !  !
        Segment already locked shared        ! ! ! ! ! ! !X! ! ! ! !X! ! ! ! !  !  !
!

     +---------------------------------------+
 7       Volume/System name incorrect         ! ! ! !X! ! ! ! ! ! ! ! ! ! ! ! !  !  !
!
        Master processor overload             !X! ! ! ! !X! ! ! !X! ! !X! ! ! ! X!  !
!
        Bad password format                   ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !X!  !  !
!

     +---------------------------------------+
                                               0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11
12
```

(Continued on next page)

Request code

Error Code	Abbreviated Message Text	ORWCCORSRODLGFFA OSG PERALCLEODIOFLUPDRS EAITOAOOPCOCNONWCES NDTSTCFEKCLSAS ETEKNKKDL	0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12
8	Permanent I/O error	!X!X!X!X!X!X!X!X!X!X!X! !X! ! !X! X! X! !	
9	Request/option not supported	!X!X!X!X!X!X!X!X!X!X!X!X!X!X!X! X! X! X!	
A	Catalog damaged !X! ! !X!X!X! ! !X! ! ! !X! ! !X! ! ! !		
B	VAU map damaged !X!X!X!X!X!X!X!X! ! !X!X!X!X!X! ! ! !		
C	No FPS space	!X! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !	
D	System not accessible	!X! ! ! ! !X! ! ! !X! ! !X! ! !X! X! ! !	
E	System connection broken	!X!X!X!X!X!X!X!X!X!X!X!X!X!X!X! X! ! !	
F	Network overload	!X! ! ! ! !X! ! ! !X! ! !X! ! !X! X! ! !	
10	XFS RN not valid	! !X!X!X!X! !X!X!X! !X!X! !X!X! ! ! ! !	
11	XFS buf not on page boundary	! !X!X! ! !X!X! ! ! ! ! ! ! ! ! ! ! ! !	
12	XFS namestring format error	!X! ! !X! !X! ! ! !X! ! !X! ! !X! X! ! !	
13	Operation out of sequence	! !X!X!X! ! !X!X!X! ! !X! !X!X! ! ! ! !	
14	XFS operation out of sequence	! !X!X!X! ! !X!X!X! ! !X! !X!X! ! ! ! !	
15	Master processor overload	! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !X! ! ! !	

0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11
12

CHAPTER 11
REQUEST LOGGER UTILITY

The Request Logger Utility is a diagnostic tool used to generate a log file containing a record of system requests issued by a slave to an OIS master.

The Request Logger runs in one slave processor, while the program being logged runs in another. The logger acts as king, taking the other slave for direct control. Requests to the master from the pawn are then intercepted by the king. The king passes the request onto the master and posts the master's response back to the pawn. It then logs a record of the request to its screen and/or to a log file.

11.1 INVOKING THE REQUEST LOGGER

The Request Logger is available as an option on the Program Development menu and is, itself, menu-driven.

The logger's primary menu asks for a pawn unit number. This can be any valid unit number of the system. Note that it is possible to specify a slave on another system by first attaching to the remote system and then running the Request Logger from that system.

Once the logger has successfully opened the slave for direct control, it asks the user to enter a file name and a load address. The file name must be a properly formatted, but undelimited, OIS namestring specifying the name of the program file to be loaded into the pawn for subsequent logging; if omitted, the file name defaults to DOS.START, the program that loads the pawn's personal menu. The load address must be a hexadecimal value specifying where in pawn memory the program is to be loaded; if omitted, the load address defaults to x'0100'.

Once this information has been entered, the logger loads the pawn with the specified file and displays the message "Program Loaded".

Any direct control errors encountered while reading or writing to the pawn are fatal; the request logger displays an error message, releases the pawn from direct control, and returns to its primary menu. (Any other errors detected during normal processing result in the display of an appropriate error message; processing continues as usual.)

11.2 CHOOSING LOG OPTIONS

Once the program is loaded into the pawn, the logger displays its options menu. There are three sections to this menu, reflecting the three types of options that must be selected: stepping mode, request type, and log destination.

The user selects a particular option by spacing down to the appropriate line and pressing the INSERT key. When all options have been selected, the user presses the EXECUTE key to start processing.

At the bottom of its options menu, the logger also displays the names of the workstation's physical and currently attached system. Note that if the physical system is running an operating system release prior to version 7.2, double quotes are displayed for both system names.

11.2.1 Stepping Mode

The stepping mode determines how the logger will submit requests to the master from the pawn. The user can select one of three stepping modes.

All Requests

If the user selects "All Requests", the logger will transmit and record pawn requests without interruption, until the user presses the CANCEL key.

Single Request

If the user selects "Single Request", the logger will transmit and record one request at a time. After each request is recorded (to the display and/or the log file), the logger will suspend execution and prompt the user. It will not continue processing until the user presses the EXECUTE key.

n Requests

If the user selects "n Requests", the logger will ask the user to enter a number between 1 and 99 (decimal) specifying the number of requests that the logger should transmit and record before stopping. After "n" requests are recorded, the logger will suspend execution and prompt the user. It will not continue processing until the user presses the EXECUTE key.

11.2.2 **Request Types**

The user must specify the types of requests that the logger is to display (either on the screen or in a log file). There are ten request types and the user can select any number.

Immediate Requests

If the user selects "Immediate Requests", the function code and data pointer for all immediate requests will be displayed, along with the actual request data. In some cases (such as the Get Drive Vector request), the data will be displayed twice since the original request data is overwritten by the master.

Nonimmediate Requests

If the user selects "Non-Immediate Requests", the function code and RCB pointer for all nonimmediate requests will be displayed, along with the contents of both the original and returned RCBs. Depending on the request, specific information, such as the RN or unit number, will also be displayed.

Namestrings

If the user selects "Namestrings", every time the pawn issues a request that has an associated namestring (such as an Open), the namestring will be displayed.

Data Read

If the user selects "Data Read", any data read by a Read or Read Lock request will be displayed (by sectors).

Data Written

If the user selects "Data Written", any data written with a Write request will be displayed (by sectors).

Open Disk Count

If the user selects "Open Disk Count", a count will be maintained and displayed of all disks that the pawn opens for direct control after the option is selected.

Open File Count

If the user selects "Open File Count", a count will be maintained and displayed of all files that the pawn opens for access after the option is selected.

Open Slave Count

If the user selects "Open Slave Count", a count will be maintained and displayed of all slaves the pawn opens for direct control after the option is selected.

Time Requests

If the user selects "Time Requests", the current system time will be displayed before and after each request logged.

Log Blaster Input File

If the user selects "Log Blaster input file", the following request options are automatically selected: immediate requests, nonimmediate requests, namestrings, and time requests. All other request options are deselected. (This option is designed to create a log file to be used as input to an OIS system utility. Therefore, if the user selects this option, a text file must also be specified as the log file destination. Refer to Section 11.2.3.)

11.2.3 LOG DESTINATION

The user must also specify the log destination. Both the Request Logger's workstation screen and an actual log can be selected simultaneously.

If the user selects "Log", the logger asks the user to enter a namestring specifying the type and name of the log. This namestring must be either a properly formatted and delimited OIS text file name or a properly formatted, but undelimited, queue name (including a system name). A system name is not required with a text file, but if it is omitted, it defaults to the currently attached.

11.3 CHANGING OPTIONS OR TERMINATING LOGGING

Once the logger has started processing, the user can interrupt it at any time by pressing the CANCEL key. Note, however, that the logger may not respond to this immediately. Processing of the current request must complete (including the logging of any data read or written) before the logger will suspend execution.

If the CANCEL key has been pressed, the logger will suspend execution and return to its options menu once the current master request is complete. (Note that this will also temporarily halt the pawn since the logger is no longer intercepting and processing requests to the master.) At this point, the user can change the existing options and then continue logging by pressing the EXECUTE key. Alternately, the user can press the CANCEL key a second time. This causes the logger to close any open log files, release the pawn from direct control, and redisplay its primary menu.

11.4 RESTRICTIONS

While the pawn is running under the control of the Request Logger, the logger processes the following requests directly instead of passing them onto the master:

- Terminate: When the pawn issues a Terminate request, the logger attempts to close all files currently open by the pawn. It then attempts to reboot the pawn by reinitializing the pawn's page zero and reloading/executing DOS.START.

- Define File Source Connection: When the pawn issues a Define File Source Connection request, the logger will record the request (if nonimmediate requests have been selected for logging). However, it will send a failure code back to the pawn indicating that the request is unsupported.

- Supervisory Read: The logger will pass a Supervisory Read request onto the master only if the request is for master performance data. Any other type of Supervisory Read will be recorded (if nonimmediate requests have been selected for logging), but the logger will send a failure code back to the pawn indicating that the request is unsupported.

11.5 EXAMPLE

The following log file example was generated by logging the default startup file DOS.START. The options chosen were: immediate requests, nonimmediate requests, and namestrings. While passwords would normally appear in a log, they do not appear in this example.

```
GET SYSTEM CONFIGURATION
      SCA =04 POINTER = 05EC
      DATA
            0702 2008 504f 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0106 0008 0000 0000 0000

SET CHARACTERISTICS
      SCA = 14  POINTER = 0261
      DATA
            0CFF

OPEN
      SCA = 01  POINTER = 0020
      FILE = ";DOS.START17"
      RCB = 0000 0000 F901 0000 0000 0000 0000 0000
      RCB = 0001 0000 F901 0000 0000 0000 0000 0000

OPEN
      SCA = 01  POINTER = 0020
      FILE = ";DOS.START5"
      RCB = 0000 0000 F901 0000 0000 0000 0000 0000
      RCB = 0001 0000 F901 0000 0000 0000 0000 0000

OPEN
      SCA = 01  POINTER = 0020
      FILE = "DOS.MENU"
      RCB = 0000 0000 1702 0000 0000 0000 0000 0000
      RCB = 0080 ED00 1702 3000 0090 0000 0000 0000
      RN = ED

READ
      SCA = 01  POINTER = 0020
      RN = ED
      RCB = 0100 EDFF 0001 0000 0000 0000 0000 0000
      RCB = 0181 EDFF 0001 0000 0030 0000 0000 0000

CLOSE
      SCA 01  POINTER = 0020
      RN = ED
      RCB = 0400 EDFF 0001 0000 0030 0000 0000 0000
      RCB = 0480 EDFF 0001 0000 0030 0000 0000 0000

SET SLAVE ACTIVITY
      SCA = 0C  POINTER = 0822
      DATA
            00
```

```
GET SYSTEM CONFIGURATION
      SCA = 04 POINTER = 2E47
      DATA
            0702 2008 504F 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0106 0008 0000 0000 0000

GET SYSTEM CONFIGURATION
      SCA = 04 POINTER = 2E47
      DATA
            0702 2008 504F 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0106 0008 0000 0000 0000

ATTACH
      SCA = 13 POINTER = 2A78
      DATA
            8009 0000 0000 5A

GET SYSTEM CONFIGURATION
      SCA = 04 POINTER = 2E47
      DATA
            0702 2008 504F 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0106 0008 0000 0000 0000

GET SYSTEM CONFIGURATION
      SCA = 04 POINTER = 2E47
      DATA
            0702 2008 504F 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0106 0008 0000 0000 0000

ATTACH
      SCA = 13 POINTER = 2A78
      DATA
            8009 0000 0677 2F
      SYSTEM = "OZ;"

GET SYSTEM CONFIGURATION
      SCA = 04 POINTER = 2E47
      DATA
            0702 2008 504F 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0106 0008 0000 0000 0000

OPEN
      SCA = 01  POINTER = 2F06
      FILE = "DOS.MENUm"
      RCB = 0000 0000 A00C 0000 0000 0000 0000 0000
      RCB = 0080 ED00 A00C 1400 0000 0000 0000 0000
      RN = ED

READ
      SCA = 01  POINTER = 2F06
      RN = ED
      RCB = 0100 ED14 0033 0000 0000 0000 0000 0000
      RCB = 0180 ED14 0033 0000 0014 0000 0000 0000
```

```
CLOSE
      SCA = 01  POINTER = 2F06
      RN = ED
      RCB = 0400 ED00 0000 0000 0082 0000 0000 0000
      RCB = 0480 ED00 0000 0000 0082 0000 0000 0000

OPEN
      SCA = 01  POINTER = 2F06
      FILE = "OZ;MAILBOX.U23"
      RCB = 0000 0002 CB2E 0000 0000 0000 0000 0000
      RCB = 0080 ED00 CB2E 0000 0000 0000 0000 0000
      RN = ED

REOPEN
      SCA = 01  POINTER = 2F06
      RN = ED
      RCB = 0800 EDFF 0000 0000 0000 0000 0000 0000
      RCB = 0880 ED00 0000 0000 0000 0200 0000 0000

GET SYSTEM CONFIGURATION
      SCA = 04 POINTER = 0823
      DATA
            0702 2008 504F 5A00 0000 0000 0082 00EA
            FFFF FFFF FF42 0107 0008 0000 0000 0000

REOPEN
      SCA = 01  POINTER = 2F06
      RN = ED
      RCB = 0800 EDFF 0000 0000 0000 0000 0000 0000
      RCB = 0880 ED00 0000 0000 0000 0200 0000 0000

GET TIME AND DATE
      SCA = 02  POINTER = 0BB8
      DATA
            3417 0F1C 0352
```

At this point, the main menu loops doing Reopen and Get Time and Date requests until a menu field is chosen.