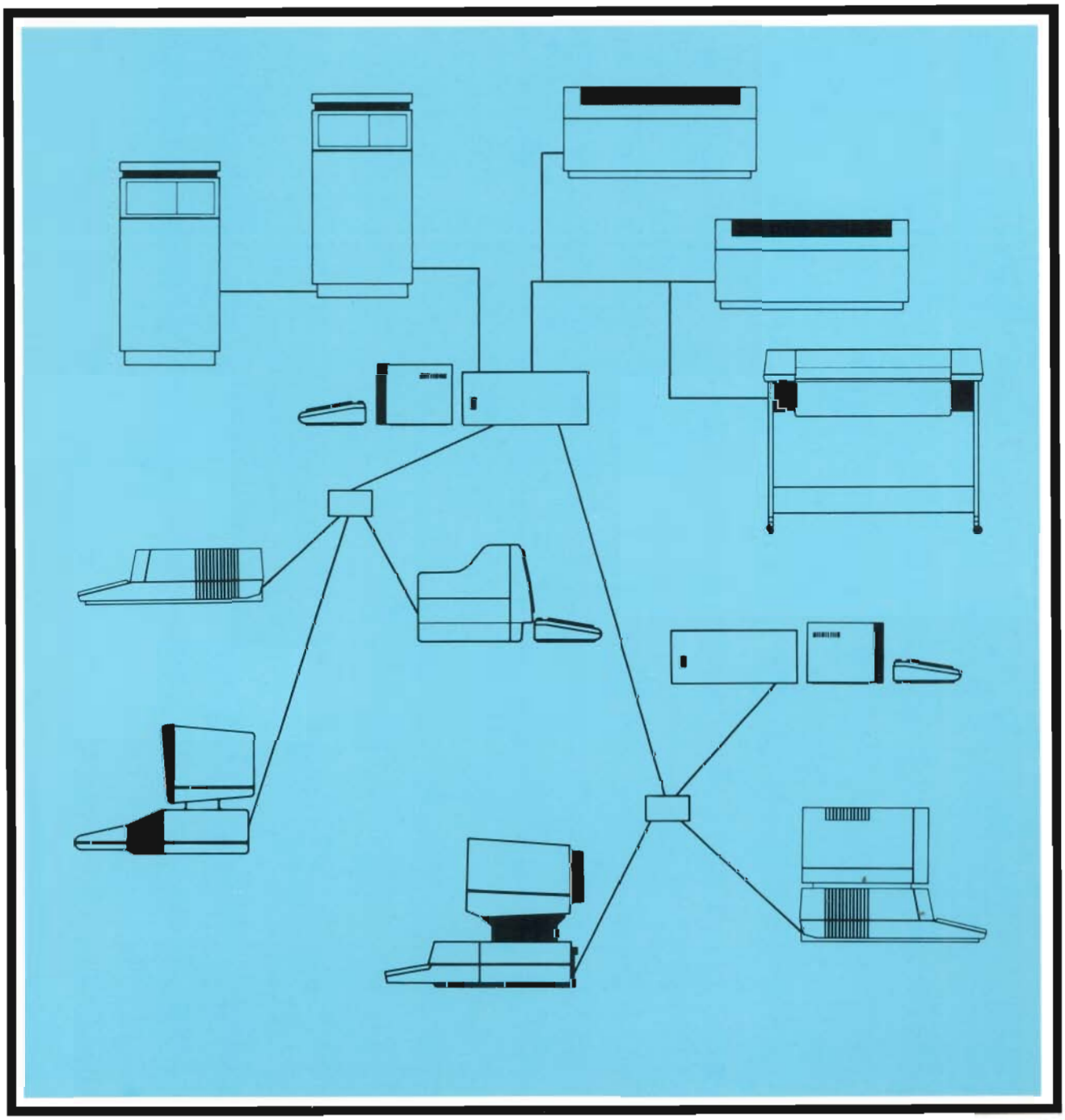
HP Computer Systems

HEWLETT PACKARD

Shared Resource Management
HP 9835/9845 Workstation Manual



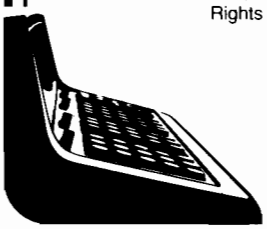
HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

Shared Resource Management

for HP 9835/9845 Workstation Manual

Manual Part No. 98619-90040



Hewlett-Packard Company
3404 East Harmony Road, Fort Collins, Colorado 80525

Printing History

New editions of this manual will incorporate all material updated since the previous edition. Update packages may be issued between editions and contain replacement and additional pages to be merged into the manual by the user. Each updated page will be indicated by a revision date at the bottom of the page. A vertical bar in the margin indicates the changes on each page. Note that pages which are rearranged due to changes on a previous page are not considered revised.

The manual printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The manual part number changes when extensive technical changes are incorporated.

January 1984...Second Edition

Table of Contents

Chapter 1

Shared Resource Programming for the HP 9835A/B and 9845B/C Computers

Introduction

It is assumed that you are familiar with programming the HP 9835A/B or HP 9845B/C computer in BASIC. The Shared Resource Management (SRM) System adds capabilities to your computer; it's not a replacement of existing capabilities. The two main purposes of the system are to provide shared access to files, and to share mass storage devices. From these purposes come the rest of the system:

- the directory structure
- protection of files and directories
- locking and unlocking of files
- extendable files
- and sharing of files.

The system makes as few changes to the operation of your workstation as possible while adding these enhancements. The main differences in workstation operation are related to the mass storage unit specifier (msus) and the specifiers used to identify files and directories. This chapter provides information needed to effectively use the shared resource system. If you have any questions concerning local operation of your computer, consult the appropriate BASIC manuals.

This chapter provides:

- an overview of the shared resource system.
- a system concepts section.
- a Language Reference section.
- a Glossary section.
- a list of appropriate Error Codes.

System Startup

To see the main areas of the shared resource system, we will perform the following operations:

- Insert the shared resource ROMs into your HP 9845 computer.
- Insert the Shared Resource ROMs into your HP 9835 computer.
- Initial System Installation (SRM)
- Catalog the root directory.
- Create your own directory in the root directory.
- Create several files in your directory, including another directory.
- Protect these new files.
- Change the capabilities assigned to a password.
- Put a file created locally into your remote directory.
- Print an ASCII file to a spooled printer.
- Read a file created by an HP 9826.
- Lock and unlock a file.
- Purge a file from your directory.
- Quit the system for the day.

If you have any problems with the concepts or terminology used in this section, see the glossary and the system concepts sections of this chapter.

Inserting the Shared Resource ROMs Into Your HP 9845B/C Computer

System installation and configuration are discussed in the SRM Hardware Installation Manual, part number 98619-90020. When your computer is properly connected to the shared resource system, it is referred to as a workstation. The firmware your computer needs to communicate with the Shared Resource Management System is the two ROMs in the Resource Management ROM, part number 98419A.

Place the ROMs in their appropriate ROM drawer locations. The black-labeled ROM, part number 09845-65585, in the right-side drawer, and the green-labeled ROM, part number 09845-65586, in the left-side drawer. Now power-up your computer normally.

That is all there is to it. Your computer is now fully operational and an integral part of the Shared Resource Management System.

Inserting the Shared Resource ROMs into your HP 9835A/B Computer

System installation and configuration are discussed in the SRM Hardware Installation Manual, part number 98619-90020. When your computer is properly connected to the shared resource system, it is referred to as a workstation. The firmware your computer needs to communicate with the system is in three ROMs in the Resource Management ROM, part number 98319A.

Place the three ROMs in any open locations in any HP 9835 ROM drawer. Now power up the computer as you normally would. That is it. Your HP 9835 is now an integral part of the Shared Resource Management System.

Cataloging the Root Directory

A quick way to see that everything is working now is to catalog the root directory. To do this, you type in:

```
MASS STORAGE IS ":REMOTE" EXECUTE
CAT EXECUTE
```

The first statement specifies the shared resource (remote) msus as the one we want to work with. This statement causes the root directory to become our working directory. The second statement produces a catalog listing as it would in local operation. The result of this statement should look similiar to the display shown below. The file names may be different for the display of your computer, but the form should be the same.

CAT of Root Directory

```
REMOTE 05,00  DIR:                    VOL:DISC1              AVAIL SPACE:    9308160
FILE                PUBLIC  SYS  FILE       FILE RECORD     LAST MODIFIED
NAME                ACCESS TYPE  TYPE       SIZE   SIZE     DATE      TIME  STATUS
=============================================================================
LP                    RW         DIR           0     24 04JUN82 06:51 AM
SYSTEMS                          DIR          24     24 04JUN82 06:53 AM
WORKSTATIONS          R          DIR         750    256 04JUN82 06:56 AM  OPENED
USERS                 MRW  9845  DIR           0    256 04JUN82 06:59 AM
```

Initial System Installation (SRM)

This section should be used if the SRM system manager is bringing the SRM system up for the first time and an HP 9835A/B or an HP 9845B/C is being used to create the SYSTEMS and USERS directories. Assuming your HP 9835/45 is properly connected to the SRM (if not, refer to the SRM Hardware Installation Manual) and is powered-up with BASIC loaded, create these two directories by typing in:

```
MASS STORAGE IS ":REMOTE"  EXECUTE
CREATE DIR "SYSTEMS"  EXECUTE
CREATE DIR "USERS"  EXECUTE
```

You may now return to the *SRM Operating System Manual* and INSTALL the SRM operating system on the system disc, beginning with Step 9 of the Initial Installation section of Chapter 3.

All workstation users should create their own directories in (subordinate to) the USERS directory. This avoids cluttering the root directory with excessive directories and files.

Creating Your Own Directory

Let us now create our own directory in the USERS directory. For the purpose of this example, let's call our directory KRISTY. To create this directory, just type in:

```
CREATE DIR "USERS/KRISTY"  EXECUTE
```

To see that it was created, type in:

```
CAT "USERS"  EXECUTE
```

Your new directory is now listed in the catalog listing.

CAT Showing New Directory

```
REMOTE 05,00  DIR:                    VOL:DISC1              AVAIL SPACE:   9307648
FILE                  PUBLIC  SYS  FILE       FILE RECORD    LAST MODIFIED
NAME                  ACCESS TYPE  TYPE       SIZE   SIZE    DATE     TIME  STATUS
=============================================================================
DERALD                             DIR          24     24 04JUN82 06:53 AM
PROJECT_ALPHA           R    9845  DATA          0    256 04JUN82 06:59 AM
KRISTY                MRW         DIR           0     24 04JUN82 07:32 AM
```

You may get an error 62. This means that the USERS directory is protected. If this happens, go to the person in charge of your system and get the proper password to create your directory. Once your directory is created, you can freely build your file structure.

Creating Files in Your Directory

To create files in our new directory, we need to "move" to that directory. To do this, we use the MSI statement. Just type in:

```
MASS STORAGE IS "KRISTY" EXECUTE
```

To see that we have a new working directory (or moved), do a catalog listing. Note that the directory name is now KRISTY and that there are no files in the listing.

CAT of Directory KRISTY

```
REMOTE 05,00  DIR:KRISTY            VOL:DISC1            AVAIL SPACE:   9308160
FILE                PUBLIC  SYS  FILE       FILE RECORD    LAST MODIFIED
NAME                ACCESS TYPE  TYPE       SIZE   SIZE    DATE      TIME  STATUS
================================================================================
```

Let's first create an ASCII file named JOB1 with an extent size of 100 records. To do this, type in:

```
CREATE ASCII "JOB1",100 EXECUTE
```

Let's next create a directory file called PROJECT_1. Note that this is a fixed length file and so no extent size is specified. Type in:

```
CREATE DIR "PROJECT_1" EXECUTE
```

And finally, let's create another ASCII file called COMMON with extent size 100. This file will be the one containing information we wish to share with other users of the system. Just type in:

```
CREATE ASCII "COMMON",100 EXECUTE
```

To see that all of these files were properly created, do a CAT.

CAT of Directory KRISTY

```
REMOTE 05,00  DIR:KRISTY            VOL:DISC1              AVAIL SPACE:   9255936
FILE                PUBLIC  SYS  FILE       FILE RECORD    LAST MODIFIED
NAME                ACCESS TYPE  TYPE       SIZE   SIZE    DATE      TIME  STATUS
=============================================================================
JOB1                  MRW        ASCII         0    256 04JUN82 07:35 AM
PROJECT_1             MRW        DIR           0     24 04JUN82 07:36 AM
COMMON                MRW        ASCII         0    256 04JUN82 07:36 AM
```

Let's also create an ASCII file under the PROJECT_1 directory. To do this, type in:

```
CREATE ASCII "PROJECT_1/PURGEME",100 EXECUTE
```

You do not have to "move" to a directory each time you use that directory. The use of a directory path (a list of directories, separated by slashes, going from either the root directory or current working directory to the desired directory) as a part of the file specifier allows you to work with a directory or file without "moving." See the Language Reference section of this chapter for many examples of this directory path feature.

Protecting Your Files

Let's first protect our directory file so that others will not be able to change anything in it. This is done with the PROTECT statement. We wish to remove both MANAGER and WRITE capability from the public domain. This will allow others to CAT our directory, but not to change anything in it. Our password will be "PASSME". Since you cannot protect an open directory, we must first close KRISTY. To do this, we type in:

```
MSI ",,"  EXECUTE
PROTECT "KRISTY", ("PASSME":MANAGER,WRITE)  EXECUTE
```

The first statement above "moves" us to the directory immediately superior to KRISTY. Moving out of a directory closes it. In this case, it would be the USERS directory. To see that the capabilities for KRISTY is changed, do a CAT. Notice that the PUBLIC ACCESS column now shows a single R in the line for KRISTY.

CAT Showing Protection of KRISTY

```
REMOTE 05,00  DIR:   USERS          VOL:DISC1              AVAIL SPACE:    9255936
FILE                PUBLIC  SYS  FILE       FILE RECORD    LAST MODIFIED
NAME                ACCESS TYPE  TYPE       SIZE   SIZE    DATE      TIME  STATUS
================================================================================
DERALD                           DIR          24     24 04JUN82 06:53 AM
PROJECT_ALPHA          R   9845  DATA          0    256 04JUN82 06:59 AM
KRISTY                 R         DIR          72     24 04JUN82 07:36 AM
```

Let's next remove all capabilities from our file JOB1. To do this, we must first "move" to the directory KRISTY and then use the PROTECT statement. Our password will be "EXCLUSIVE". Type in:

```
MASS STORAGE IS "KRISTY<PASSME>:REMOTE"  EXECUTE
PROTECT "JOB1", ("EXCLUSIVE":MANAGER,WRITE,READ)  EXECUTE
```

To protect the MANAGER capability of PROJECT_1, type in:

```
PROTECT "PROJECT_1",("MGR":MANAGER)  EXECUTE
```

We want everyone to be able to read and write to our COMMON file, so we will leave it alone.

To see the results of our efforts, do a catalog listing of the directory KRISTY.

CAT of Directory KRISTY

```
REMOTE 05,00  DIR:KRISTY            VOL:DISC1              AVAIL SPACE:   9255936
FILE                  PUBLIC  SYS   FILE        FILE RECORD     LAST MODIFIED
NAME                  ACCESS TYPE   TYPE        SIZE   SIZE     DATE      TIME  STATUS
=============================================================================
JOB1                                ASCII          0    256 04JUN82 07:35 AM
PROJECT_1               RW          DIR            0     24 04JUN82 07:36 AM
COMMON                  MRW         ASCII          0    256 04JUN82 07:36 AM
```

To protect the READ and WRITE capabilities of PURGEME, the file we created under PROTECT_1, type in:

```
PROTECT "PROJECT_1/PURGEME",("PASSME":READ,WRITE)
```
EXECUTE

To see that this file is now protected, type in:

```
CAT "PROJECT_1"
```
EXECUTE

CAT of Directory PROJECT _1

```
REMOTE 05,00  DIR:PROJECT_1         VOL:DISC1              AVAIL SPACE:   9249792
FILE                  PUBLIC  SYS   FILE        FILE RECORD     LAST MODIFIED
NAME                  ACCESS TYPE   TYPE        SIZE   SIZE     DATE      TIME  STATUS
=============================================================================
PURGEME                 M           ASCII          0    256 04JUN82 07:45 AM
```

Putting a Local File on the System

On the HP 9835/9845, program files created locally must be re-compiled for shared resource operations. You do this by first LOADing the local program. Then you SAVE it. The saving process converts the program into a DATA file format. Next you GET the file. And finally, you STORE the file on the shared resource system. For example, if the PROG file named PROG1 was to be transferred from local media ":T15" to the shared resource system, you would type in:

```
LOAD "PROG1:T15" EXECUTE
SAVE "PROG1T:T15" EXECUTE
GET "PROG1T:T15" EXECUTE
STORE "KRISTY<PASSME>/PROG1:REMOTE" EXECUTE
```

If you then did a CAT of KRISTY, you would see that the file PROG1 was now in that directory. A CAT would look like the following example.

CAT of Directory KRISTY

```
REMOTE 05,00  DIR:KRISTY            VOL:DISC1            AVAIL SPACE:   9224192
FILE                PUBLIC  SYS  FILE       FILE RECORD    LAST MODIFIED
NAME                ACCESS TYPE  TYPE       SIZE   SIZE    DATE      TIME  STATUS
================================================================================
JOB1                             ASCII         0    256 04JUN82 07:35 AM
PROJECT_1             RW         DIR          24     24 04JUN82 07:45 AM
COMMON               MRW         ASCII         0    256 04JUN82 07:36 AM
PROG1                MRW  9845   DATA        510    256 04JUN82 07:49 AM  OPENED
```

To transfer all other file types (non-PROG files) from a local device to a remote device, use the COPY statement. For example:

```
COPY "DATA:T15" TO "KRISTY<PASSME>/DATA1:REMOTE" EXECUTE
```

Printing to the Spooler

Let's assume that a spooler directory (spooler) named "LP" has been created. Details concerning the creation of spoolers are found in the "Interfaces and Peripherals" chapter of the SRM Operating System Manual, part number 98619-90030. To access the spooler's printer, simply put the file you want printed into the spooler directory. For example, to list your program to the printer, execute:

```
SAVE ASCII "LP/FILENAME:REMOTE" EXECUTE
```

The above statement uses that same syntax used to save ASCII files in any other directory. To print the contents of an existing ASCII file, execute:

```
COPY "MYFILE:REMOTE" TO "LP/MYFILE:REMOTE" EXECUTE
```

If you desire to print out data (something other than a program listing), you can create an ASCII file and then PRINT # the data into the file. For example:

```
MSI ":REMOTE" EXECUTE
CREATE ASCII "/LP/PERFORMANCE",100 EXECUTE
ASSIGN #1 to "/LP/PERFORMANCE" EXECUTE
PRINT #1;"    THIS IS THE DATA TO BE PRINTED" EXECUTE
ASSIGN #1 to * EXECUTE
```

The system will automatically wait until a file sent to the spooler is non-empty and closed before printing begins. The ASSIGN #1 TO * closes the example file, and so would initiate the printing.

To see what the spooler is doing, CATalog the spooler directory. The file currently being printed is listed as LOCKED. Files currently being written to the spooler (either printer or plotter) are listed as OPENED.

In contrast to the SRM 1.0 system, the SRM 2.0 operating system allows non-ASCII files to be sent to the printing device as a byte stream.

Note

Any non-ASCII file sent to the spooler is printed exactly as the byte stream sent. Unless the user sets up his non-ASCII file correctly, improper printer output or operation could result. Therefore, it is recommended that the workstation operator use only ASCII type files when spooling to a printer or plotter.

Since ASCII files permit only string data; to print numeric data, convert that data into strings using the VAL$ function before printing. Each string will be printed on a separate line because carriage-return line-feed is inserted after each string. To put several strings on one line, concatenate them into one string before printing them to a file. You can insert ASCII control characters anywhere in the data by using the CHR$(n) statement. The spooler has one option for controlling formfeeds: by prefixing your file name with FF, the spooler will do a formfeed for each page to avoid printing on the paper's perforations. For example:

```
SAVE ASCII "/LP/FF_MYTEXT" EXECUTE
```

The spooler will automatically print a one page header for each file printed. This header will contain the directory path and filename, the date, and time of the printing.

To abort a printing in progress, you can either take the printer off-line or put the spooler in the DOWN state from the shared resource controller. When the spooler is placed in the UP state and the printer is on-line, the whole file will automatically be reprinted. If you do not want that file printed, use the PURGE command.

Getting a File Created on an HP Series 200

Let's assume that a user of an HP Series 200 workstation has an ASCII file that we need to read. Unless the BDAT file Binary is utilized, ASCII files are the only files that can be shared between different workstation models. This user has a directory called KAREN located in the USERS directory with the password KLSVE assigned to it. The password KLSVE protects the READ capability. The file we need to read is called COMMON and is located under the directory KAREN. COMMON has no passwords assigned to it. To gain access to a program in that file, type in:

```
GET "KAREN<KLSVE>/COMMON:/USERS"  EXECUTE
```

That is all there is to it.

LOCK and UNLOCK a File

To obtain sole access to a file, the LOCK and UNLOCK statements are used. For this example, assume that a critical operation must be performed on the file AFILE and for the time it takes to do that operation, the file must be locked. The program would look like the following:

```
1000 ASSIGN "AFILE" to #1
1010 LOCK #1
1020 ! Begin critical process
1030 READ #1;A
1040 IF A>0 THEN
1050 PRINT #1;B
1060 ! End critical process
1070 UNLOCK #1
```

For further information, see the LOCK and UNLOCK statements in the Language Reference section of this chapter.

Purging a File

For this example, let's assume that the path from the root directory to the file we wish to purge is USERS/KRISTY/PROJECT_1 and the file to be purged is PURGEME with password PASSME. PASSME protects the MANAGER capability. PROJECT_1 has its WRITE capability protected with the password MGR. To purge the file, type in:

```
PURGE "USERS/KRISTY/PROJECT_1<MGR>/PURGEME<PASSME>"  EXECUTE
```

Purge works just as it does in local mode, but the whole path to the desired file must be specified including any needed passwords.

Quit the System for the Day

At the "end of the day", you need to close the shared resource files you have been using. If you do not close these files, the shared resource system's performance may be degraded. Close files as you normally would. To close your working directory, use the MSI":T15" statement. If you have any doubts, the SCRATCH A statement will completely release your access to the system. Resetting your workstation with (CTRL)-(STOP) will close all files except your working directory.

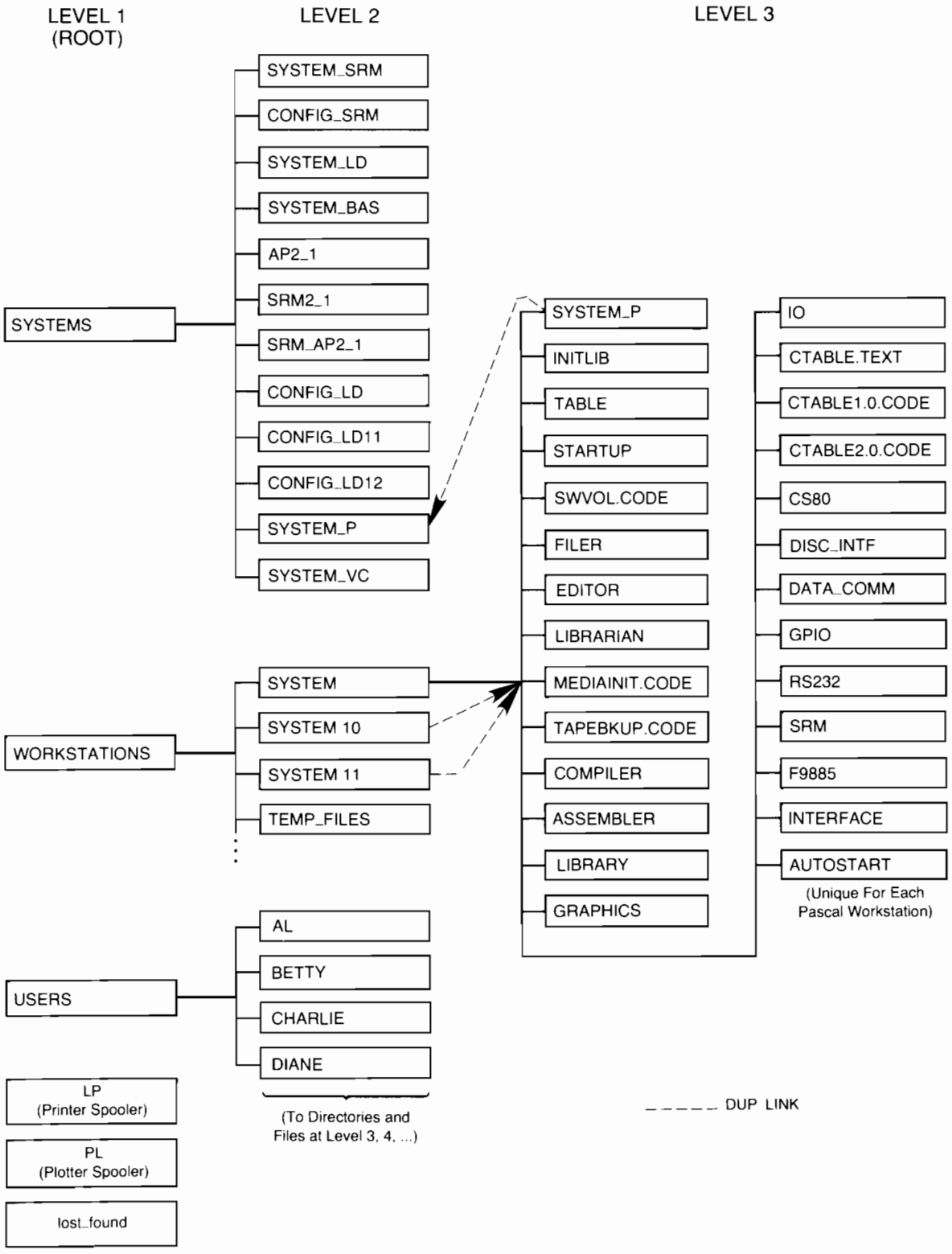
Chapter 2
System Concepts

The following is a detailed look at some of the unique concepts and structure used with your Shared Resource Management System. These concepts include:

- Directory structure.
- Specifying files and directories.
- File and directory names.
- Mass Storage Unit Specifiers.
- Accessing remote files.
- Restricted access to files and directories.
- Passwords.
- Shared access to files
- Extending files.
- Mass Storage Support

Directory Structure

The Shared Resource Management System uses a hierarchical directory structure to organize files on the disc. A directory is a fixed-record-length (24 bytes), fixed-format file that you use to organize and control access to other files. Directories are created with the CREATE DIR command. They contain system information about the files directly beneath them. Every file in your system must be "under" a directory. Since directories themselves are a type of file, they must also be under other directories. The system starts with the "root directory". All other files and directories are at level 1, 2, etc. You cannot create any directory at a higher level than the first level. The illustration on the following page shows a recommended directory structure for an SRM system using both BASIC and Pascal workstations.



Example of an SRM Directory/File Structure

Directories can be:

- Created with the CREATE DIR statement. The location of the new directory within the hierarchical structure is specified when it is created.
- Cataloged with the CAT statement.
- Renamed with the RENAME statement.
- Protected with the PROTECT statement.
- Filled with subordinate files and directories using the CREATE, CREATE DIR, SAVE, COPY and STORE statements. Each file or directory uses one directory record. Since all shared resource files are extendable, you can add as many files to a directory as you like, limited only by the size of the disc.
- Opened and closed with the MASS STORAGE IS (MSI) statement. When the MSI statement specifies a given directory, that directory is opened. When another directory is specified with the MSI statement, the previously opened directory is closed.
- Emptied by removing all subordinate files and directories with the PURGE statement.
- Purged with the PURGE statement. Only closed, empty directories can be purged.

Directories serve two main purposes. First, they allow you to organize files. If, for instance, you're managing several projects simultaneously, you can create a directory for each project. Under each project directory, you can have a directory for each person working on the project plus directories of shared files available to all project members. Each project and each part of a project can have its own access protection to assure project security. Since files with different locations in the directory structure can have the same file name, you can use the same generic name to identify similar project functions in different projects. This makes it harder to lose files or accidentally access the wrong file.

The second purpose of directories is to help restrict access to specified groups of files. Access to directories can be restricted just as with other files. The SRM System provides two levels of file security. The directory structure allows you to restrict access to a large number of files at once. By putting selected files that you want protected under a single directory with restricted READ access, you automatically restrict the access to those files. If further protection is needed, each file can also be separately protected.

Next, let's look at how the files and directories are specified in this structure.

Specifying Files and Directories

In order to access a file, a directory file or a data file, you must specify the file's location in the hierarchical directory structure. This location is specified with a list of directory names that you must "go through" in order to reach the desired file. Each directory name is delimited by a slash (/). You can start either from your working directory or from the root directory. Your working directory is the directory specified with the most recent MASS STORAGE IS statement.

To access a file above your working directory or on a different branch, you must either use a full specifier beginning with the root directory or the double-period "directory name". The double-period ("..") means "the immediately superior directory". If your current working directory is on level 5 and you want to "move" to level 3, type in `MASS STORAGE IS "../.."`. To access a file below your working directory, you use the specifier starting with the appropriate directory

subordinate to your working directory. If the file you want to access is in your working directory, then the specifier is simply the file's name. To specify your working directory, in order to include a password that you forgot the first time (as an example), you can use the single period specifier. For example MSI ".<PASSWORD>" will specify the current working directory and also provide the access capabilities associated with the password PASSWORD.

You specify the root directory in one of two ways. A slash (/) as the first character in a specifier means to start "moving through the directory structure" beginning with the root directory. The leading slash applies only when a previous MSI ":REMOTE" statement has been executed (you are in the remote system). The slash must be part of the specifier. The second way is to terminate the specifier with the REMOTE msus. It is acceptable, but redundant, to use both root directory indicators in the same specifier.

If you're not sure what your current working directory is, execute a CAT command. CAT will tell you what your working directory is and what files are in it. You can catalog directories other than your working directory by using a specifier that designates the desired directory.

File and Directory Names

Shared resource system file and directory names can be from 1 through 16 characters long. The character set allowed for these file names is shown below:

Legal Name Characters

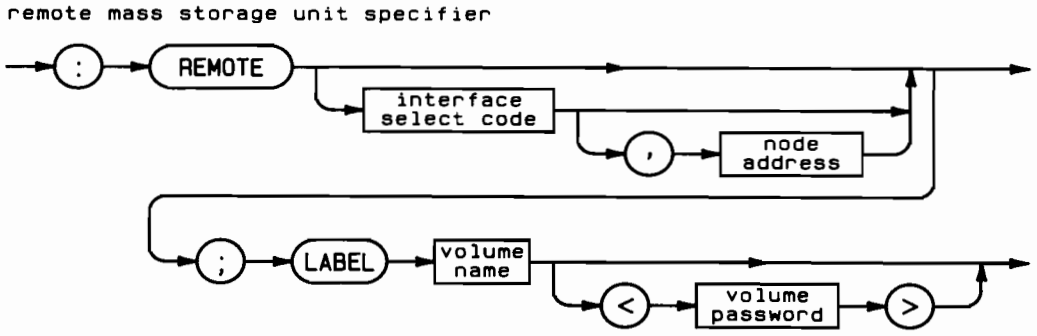
Uppercase letters	A-Z	ASCII	65-90
Lowercase letters	a-z		97-122
Foreign characters			161-254
Digits	0-9		48-57
Underscore	_		95
Period	.		46

When transferring files from local media to the shared resource system or from the shared resource system to the local media, attention needs to be paid to the rules for file names. In all cases, blanks are ignored.

Mass Storage Unit Specifiers

One main purpose of the SRM System is to provide shared access to mass storage devices. This is accomplished with the remote msus. This new syntax is shown below. Any path from the entry line on the left to the exit arrows on the right will generate a correctly syntaxed statement for the msus specifier.

Remote MSUS



Restricted Access to Files and Directories

The SRM System offers three levels of access capability for files and directories: READ, WRITE, and MANAGER. Capabilities are protected by passwords. Each password can be assigned one or more capabilities. Capabilities are assigned by specifying the following secondary words in a PROTECT statement (see the Language Reference section of this chapter for more information about using the PROTECT command). A file can have any number of password/capability pairs assigned to it.

READ — READ capability for a file allows a user to read the file (LOAD, READ #, etc.). Read capability for a directory allows a user to read the filenames in the directory (CAT), and to "go through" the directory with a statement using a file or directory specifier. For example, if READPASS protected READ on directory DIR1, then MASS STORAGE IS "DIR1<READPASS>/DIR2/FILENAME:REMOTE" would "go through" to the file FILENAME.

WRITE — WRITE capability for a file permits a user to write to the file (PRINT #, MAT PRINT #). Write capability for a directory allows the user to add or delete filenames to the directory (PURGE, CREATE, RENAME, SAVE, STORE, etc.).

MANAGER — MANAGER capability exists in two forms, public and password. Public manager capability is when the file is not manager protected. It allows the user to PROTECT, PURGE, CAT PROTECT and RENAME the file. Password manager capability is when the file is manager protected (a valid password is specified with the MANAGER capability). It gives the user all three capabilities, MANAGER, READ and WRITE.

Passwords

Passwords are string expressions from 1 thru 16 characters in length containing any ASCII character except ASCII character 62 (>). Any spaces (ASCII 32) will be removed from the password by the system.

Capabilities are assigned to passwords with the PROTECT statement. You specify a password in a file or directory specifier by enclosing it in <> symbols immediately following the PROTECTed file or directory name. For instance, if the file TEST had restricted READ access and the password is PASSME, you would enter the LOAD command as follows:

```
LOAD "TEST<PASSME>"
```
EXECUTE

If you don't specify the correct password when it is required, the system will return an error message. The following table lists BASIC keywords and the access capability needed to use them. The "Superior Directory" column tells you what access capability you need to use the commands on the directory that is superior to the file being accessed. Some commands apply only to files, some only to directories, and some to both. Commands not listed in the chart can be executed regardless of PROTECTed capabilities. Remember that MANAGER capability includes all capabilities when assigned to a password.

Capabilities Required

Dash (–) means "does not apply"

Keyword	File or Directory	Superior Directory
ASSIGN #	one or more	READ
CAT	READ	READ
CAT PROTECT	MANAGER	READ
CLOSE PLOTTER #	one or more	READ
COPY source	READ	READ
COPY destination	–	WRITE
CREATE	–	READ & WRITE
CREATE ASCII	–	READ & WRITE
CREATE DIR	–	READ & WRITE
GET	READ	READ
LINK	READ	READ
LOAD	READ	READ
LOCK #	–	–
MSI	–	READ
MAT PRINT #	WRITE	–
MAT READ #	READ	–
PLOTTER IS #	one or more	READ
PRINT #	WRITE	–
PROTECT	MANAGER	READ
PURGE	MANAGER	READ & WRITE
READ #	READ	–
RENAME	MANAGER	READ & WRITE
SAVE	–	READ & WRITE
STORE	–	READ & WRITE
UNLOCK #	–	–

Shared Access to Files

Under the SRM System, all ASCII files can be shared among the system's users. This can save disc space since only one copy of each file is needed. This one copy can then be shared among those who need access to it. On the negative side, however, it raises the danger of two users trying to write to the same part of a file simultaneously. This can cause unpredictable results since the operating system accepts information as message packets on an "as received" basis. Likewise, if one user tries to read a part of a file while another user is writing to it, problems can occur. To avoid these situations, the system provides two new BASIC keywords – LOCK and UNLOCK. These new words are used to gain sole access to files during critical operations.

LOCK and UNLOCK

There are times when it is important to prevent users from writing to the same part of the file at the same time. For example, suppose two airline operators are connected to the same system. Operator 1 wants to read the reservation list for a flight and then enter a reservation if there is room. Operator 2 wants to do the same thing. If they both try to use the system at the same time, they might both enter reservations for the same space. To avoid this, Operator 1 can LOCK the file before reading it. This prevents Operator 2 from accessing the file until Operator 1 is done. This way there can be no confusion.

Once a file is locked, only the workstation that locked that file has access to it. It is important to remember to UNLOCK a file once you are done with it. Other users are denied access until the file is UNLOCKed. More information about these keywords can be found in the Language Reference section of this chapter.

Extending Files

Under the SRM System, files grow dynamically as data is entered into them. This means that the only limit to a file's size is the size of the disc being used. However, a single file cannot span disc volumes. File space is not allocated until the file is actually used. Creating a file, for instance, will not allocate space, but writing to a file will.

The size of each file extension is determined by the extent size parameter specified in the CREATE statement. This is both the initial size of the file and the amount that the file grows each time more space is needed.

Non-Contiguous Files

The shared resource system may fragment a file in order to fill in gaps on the disc, thus making more efficient use of the media. This process, however, is entirely transparent to the user and cannot be externally controlled. This feature eliminates the need for PACKing a volume since the operating system fills in the gaps automatically.

Mass Storage Support

There are four new commands added to the BASIC mass storage statements used by the HP 9835A/B and HP 9845B/C computers. They are:

- CREATE ASCII. This statement creates ASCII files.
- CREATE DIR. This statement creates directory files.
- LOCK. This statement provides sole access to a file or directory. This statement is needed due to the shared file capability of the system.
- UNLOCK. This statement is the partner of the LOCK statement.

There are several statements that are not supported on the system. They are:

- BUFFER #
- CHECKREAD #
- INITIALIZE
- LOAD ALL / STORE ALL
- LOAD BIN / STORE BIN
- LOAD KEY / STORE KEY
- RE-SAVE / RE- STORE
- TYP

The Language Reference section of this chapter provides syntactic and semantic information about the enhancements made to BASIC mass storage statements.

Chapter 3
Language Reference

The following section contains semantic and syntactic language information in a reference format. Each keyword is listed alphabetically. The format used is shown:

Keyword

MAT PRINT

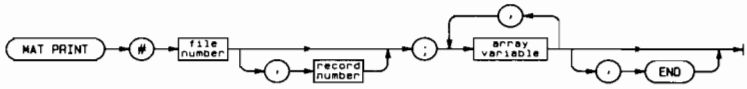
Programming Information

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN	Yes

Keyword Definition

This statement outputs data contained in array variables to DATA and ASCII files.

Syntax Diagram



Parameter Table

Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
record number	integer	1 thru 32 767
array variable		see 9845 BASIC manuals

Examples

Example Statements

```
10 MAT PRINT #2;A,END
20 MAT PRINT #10,5;Real_array,Short_array,Integer_array,String_array$
30 MAT PRINT #File_number,Record_number;Another_array
```

Semantic Information

Semantics

MAT PRINT # performs the same function to arrays as PRINT # performs for other data. See the PRINT # statement for further information. You must have WRITE capability to execute this statement.

The Syntax Diagram

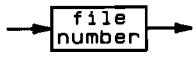
The syntax diagram is read from left to right. You follow the arrows until you exit the drawing. The statement created by any path through the diagram is a legal and executable statement.

Keywords

Keywords are indicated by the rounded-rectangular boxes. The keywords must be entered into your computer exactly as shown. Some languages allow the use of both upper- and lower-case characters. Some languages also allow any spelling from a minimal subset of the word to the full word. Special cases are described in the semantics part of the reference. Single characters can also serve as keywords. For example, MAT PRINT is a keyword shown below.

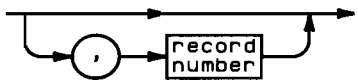
Parameters

Parameters are indicated by the square boxes. The parameter is then listed in the parameter table with description, default and range restriction information. You may be referred to a source of more complete information, such as the Glossary. The parameter "file number" is shown below.



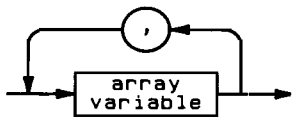
Optional Paths

When the syntax allows a choice, the diagram shows this choice by the optional path. Any path through the drawing will provide a correct statement. The optional path for specifying a record number is shown below.



Loop Paths

When a list is allowed in the syntax, a loop path is shown in the drawing. The number of times you can use the loop is specified in the semantics section of the reference. The loop to generate a list of array variables is shown below:



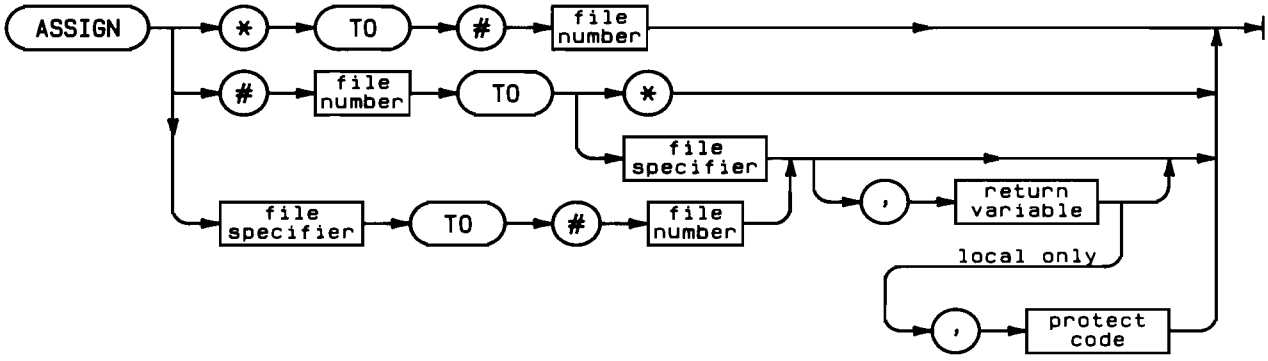
The End

The end of the drawing is shown by a short vertical line. If the drawing ends without such a line, the drawing is a part of a larger drawing. The terminator is shown below.

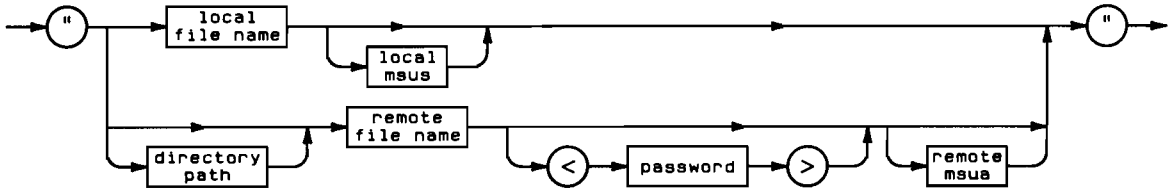
ASSIGN

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

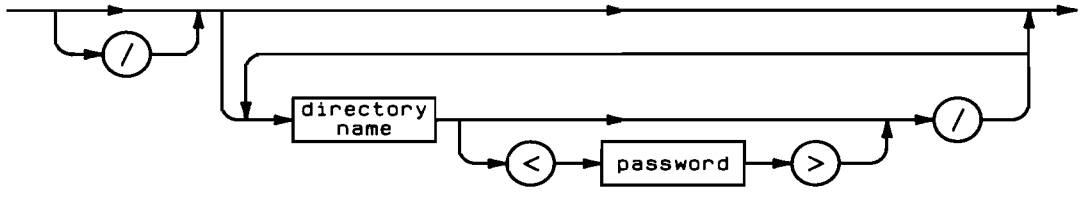
This statement allows the user to open and close specified files.



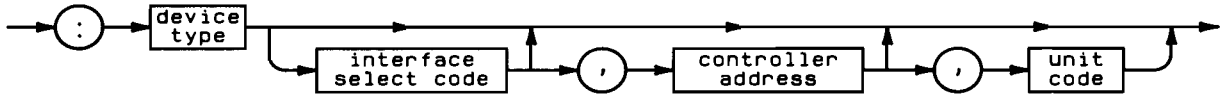
file specifier



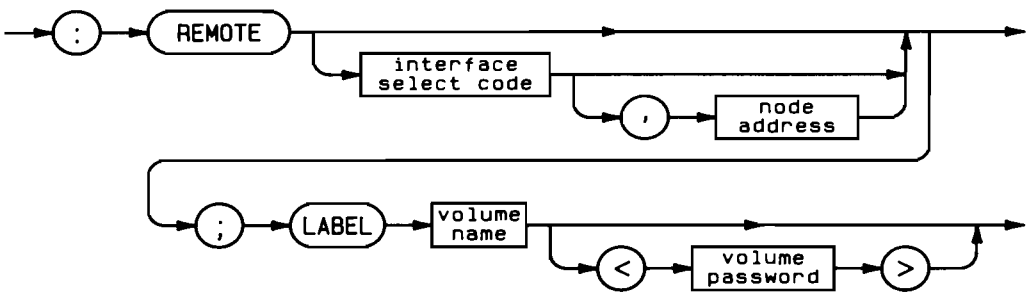
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
file specifier	string expression	(see drawing)
return variable	simple numeric variable	initial capital only
protect code	string expression	only first 6 characters used
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
password	string expression	(see Glossary)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices, and 12 for all other devices; remote: integer constant assigned to the workstation's 98609 interface. Default = 5	(see Glossary)
controller address	specifies a disc drive controller	0 thru 7
unit code	integer designating disc unit; Default: 0	0 thru 7
node address	integer constant assigned to the controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
ASSIGN #4 TO "/Dir1/Dir2<passme>/Test",Return1
ASSIGN #10 TO "Dir1/Dir2<passme>/Test:REMOTE",Return1
ASSIGN "/Eric/Kristy/Project_1" TO # 1
ASSIGN "Karen/Kathy/Project_Upgrade" TO #7
ASSIGN #4 to *
ASSIGN * to #10
```

Semantics

The first four example statements above open the specified files. The last two statements close files previously assigned.

The ASSIGN statement will open or close DATA files in local mode and both ASCII and DATA files when using the shared resource (remote) system. Remote files are always opened in shared mode. That is, other users can ASSIGN that same file. To have sole access to a file, you must LOCK it. When closed, all files are automatically UNLOCKed.

You must have READ capability on the file's directory and at least one capability for the file to be assigned for this statement to execute.

The ASSIGN statement sets up an internal files table and allows you to utilize files with the PRINT # and READ # statements. There is one table for each program environment (main program and each subprogram). Each table has room for ten entries. All entries are cleared on the next remote access when a program is RUN, and when SCRATCH, SCRATCH V, SCRATCH C, or SCRATCH A is executed. Powering up the system or executing RESET also clears these entries immediately.

The ASSIGN statement creates one file pointer when using the shared resource system. When first ASSIGNed, it points to the beginning of the file. It always points to the file location to be read or written next. The second pointer is created by the resource management controller and points to the logical end of the file. The first pointer moves anytime you write or read from the file. The second pointer moves only when you write past the end of the file. The controller's pointer will indicate the current size of the file. If you overwrite a file, the controller's pointer points to the highest point written to in the file. To move the controller's pointer to the same location as the first pointer, use the END keyword with a PRINT # statement.

All files must be ASSIGNed to a file number (opened) before referencing them with a READ # or PRINT # statement.

The optional return variable, a simple numeric variable, is set after execution to indicate various results. You can use its value to check for errors. The variable will return 0,1, or 2 as shown below. Any other error will be handled normally.

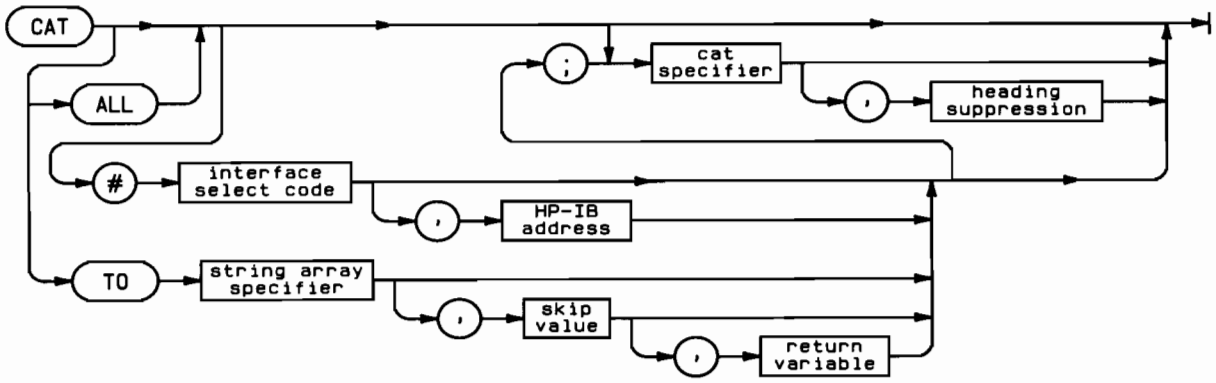
Return Variable	Meaning
0	File available
1	No such file found
2	File is protected, or wrong file type

ASSIGNed files tie up system resources and should be closed when you are finished with them.

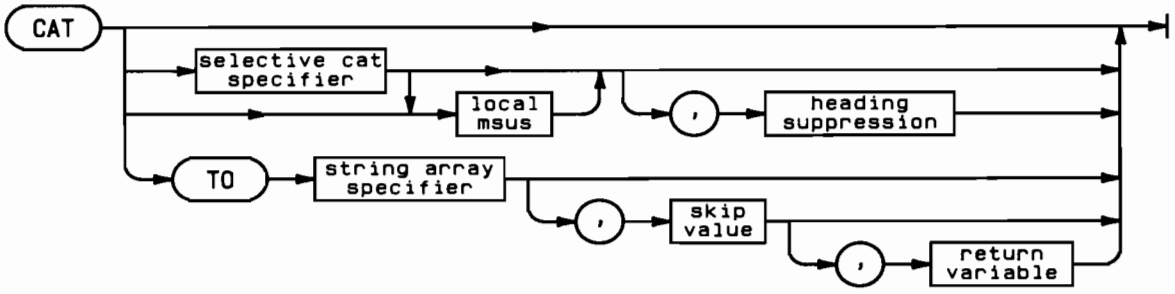
CAT

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

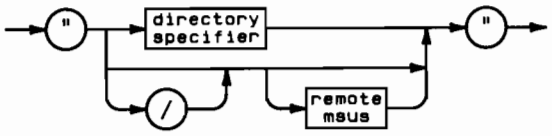
This statement outputs a listing of directory information for a specified directory.



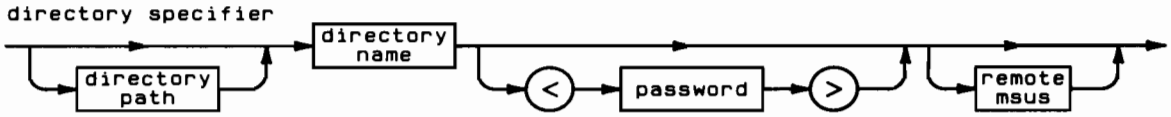
local only



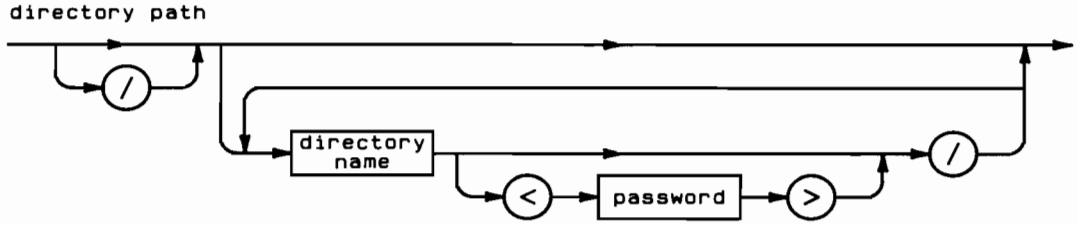
cat specifier



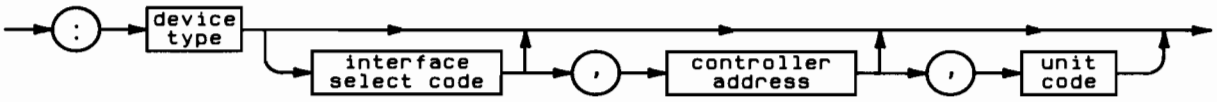
directory specifier



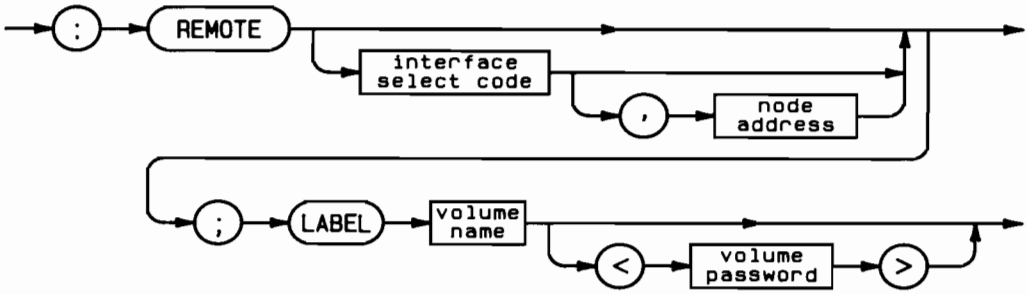
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
cat specifier	string expression	(see drawing)
heading suppression	simple numeric; Default: For CAT and CAT ALL: Heading is printed. For CAT TO $: Heading is suppressed	any simple numeric
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices. remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
HP-IB address	integer	0 thru 30
string array specifier	string array dimensioned to at least 80 characters per element for remote and at least 41 characters per element for local statements.	any valid string array
skip value	simple numeric; Default: 0	positive numerics
return variable	simple numeric variable	
directory specifier	string expression	(see drawing)
msus	string expression	(see drawing)
selective cat specifier	string expression	(see HP 9845 BASIC manuals)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)

Example Statements

```
CAT
CAT ALL
CAT TO A$(*)
CAT #0
CAT "Dir2/Dir2<passme2>:REMOTE"
CAT "A:T15"

DIR$ =":REMOTE"
CAT DIR$
```

Semantics

CAT lists all files in the directory specified. If the keyword ALL is included in the statement, the system will catalog all subordinate directories to a maximum of six levels deep in the directory tree structure. Directories for which READ capability is not available will stop the cataloging of that branch.

You must have READ capability in both the directory you want to CAT and its immediately superior directory for this statement to execute.

If you want the catalog information sent to specific local device, specify the device's select code and HP-IB address. The HP-IB address is only necessary if there is more than one device on the specified HP-IB line. If you don't specify an interface select code, the system will always send the CAT to the current PRINTER IS device. The default is the current PRINTER IS device.

If you want to send catalog information to a string array, use the CAT TO format. Each element in the string array must be dimensioned to at least 80 characters. The **skip value** specifies the number of directory entries to be ignored before the entries are listed to the array. For example, if the skip value were 3, the system would ignore the first 3 directory entries and list the fourth entry to the first element of the array. The `return variable` tells you the number of the last entry successfully listed to the array. If, for instance, your array wasn't big enough to hold the entire CAT, the return variable would contain the number of the last line entered into the array. If the return variable equals 0, then all entries in the directory have been transferred. Strings that are not written become null strings.

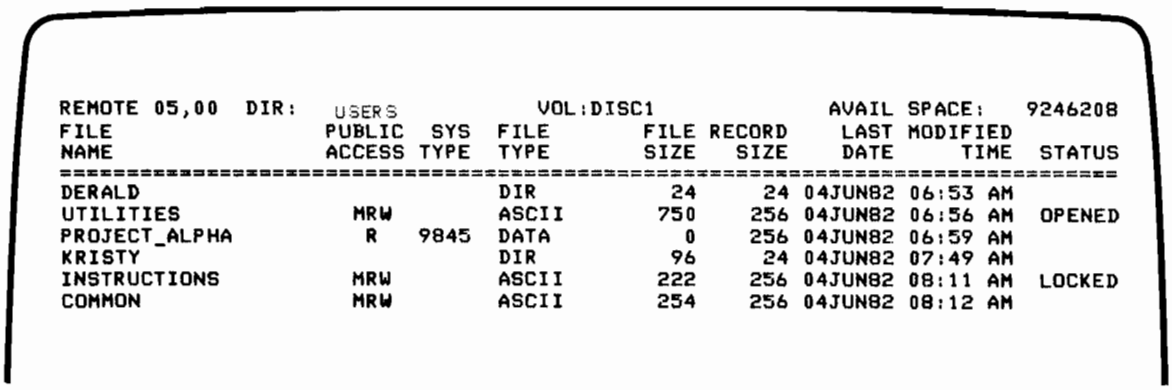
The catalog heading gives the directory name, the volume name, the interface select code, the node address and the available space on the volume. If the heading suppressor is equal to one (1) the heading is suppressed. Any other number will allow the heading to be printed.

For local CAT statements, the selective specifier is used to generate a catalog of all files that begin with the string indicated. For example:

```
CAT "A:T15" EXECUTE
```

This statement will list all files that begin with the letter "A." This feature works only on local mass storage devices.

CAT Display Example



```
REMOTE 05,00  DIR:  USERS          VOL:DISC1              AVAIL SPACE:   9246208
FILE                PUBLIC  SYS  FILE       FILE RECORD    LAST MODIFIED
NAME                ACCESS TYPE  TYPE       SIZE   SIZE    DATE     TIME  STATUS
================================================================================
DERALD                           DIR          24     24 04JUN82 06:53 AM
UTILITIES             MRW        ASCII       750    256 04JUN82 06:56 AM  OPENED
PROJECT_ALPHA          R   9845  DATA          0    256 04JUN82 06:59 AM
KRISTY                           DIR          96     24 04JUN82 07:49 AM
INSTRUCTIONS          MRW        ASCII       222    256 04JUN82 08:11 AM  LOCKED
COMMON                MRW        ASCII       254    256 04JUN82 08:12 AM
```

FILENAME — Gives the names of the files contained in the directory.

PUBLIC ACCESS — The password column tells you which capabilities ACCESS are public: M for MANAGER, R for READ, and W for WRITE. Any capabilities not listed require a password.

SYS TYPE — This column tells you what type of system the file was created on. SYS TYPE is not given for ASCII and DIR type files since they are the same on HP Series 200 and HP 9835/45 workstations.

FILE TYPE — Common file types are:

ASCII — ASCII files are files that have been saved with a SAVE ASCII statement or created with a CREATE ASCII statement. In ASCII files, every data item is represented by its ASCII value. These files have the advantage of being transportable between different SRM workstations.

BDAT — HP 9835/45 binary DATA files are not usable on SRM. Access to Series 200 BDAT files is possible with use of the BDAT binary.

BIN — A file stored in BINary code. Binary files cannot be created or accessed by the HP 9845 workstations.

DATA — The HP 9845 DATA files are files of Binary Coded Decimal formatted information plus strings. The HP 9826/9836 workstations cannot use these files.

DIR — DIRectory files are special files used to organize and control the hierarchical file system used with SRM. They must be created with the CREATE DIR command. Access is gained with the MSI statement. You cannot store, read or write data in a DIR file.You can, however, CATalog, CREATE, PURGE and PROTECT a DIRectory file.

PROG — PROG files are BASIC program files that have been stored in their compiled form.

???? A file type unknown to the system will be listed by a decimal code.

All files created by any shared resource workstation are recognized by the HP 9845 when using the shared resource system. If a file was created on an HP 9826 or HP 9836 workstation, the CATalog statement will inform you. If the file type is not recognized at all (e.g., it was created on a system not supported by shared resource), then a numeric code is displayed in the file type column.

FILE SIZE The size in bytes of the file. Local statements show the size in **records** while remote statements show the size in **bytes**.

RECORD SIZE This is the size of the defined record. The default is 256 bytes for files and 24 bytes for directories.

LAST MODIFIED DATE This is the date the file was last modified.

LAST MODIFIED TIME This is the time the file was last modified.

STATUS This column gives file status. The possible file enteries in this column are:

- OPENED A user has ASSIGNed the file, or has opened a working directory with a MASS STORAGE IS statement.
- LOCKED Some user on the system has LOCKed the file, or a SAVE or STORE is in progress.
- CORRUPT File is corrupt. The disc was powered down during a critical operation. Condition of file is unknown. Run DSCK utility to attempt file recovery. If file is lost, use latest backup.
- blank The file is closed.

File Table for Local and SRM Use

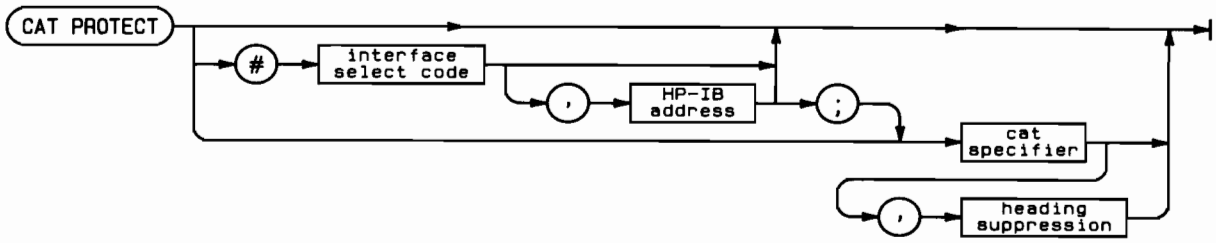
The following table shows which file types work on certain local or remote applications:

File Type	HP 9835/45 Local	HP 9835/45 Remote	HP Series 200 Remote
ASCII	No	Yes	Yes
9835/45 DATA	Yes	Yes	No
9845 BDAT	Yes	No	No
Series 200 BDAT	No	HP 9845 only w/ "BDAT" binary	Yes
9835 BIN	Yes	No	No
9845 BIN	Yes	No	No
Series 200 BIN	No	No	Yes
9835 PROG	Yes	Yes	No
9845 PROG	Yes	Yes	No
Series 200 PROG	No	No	Yes

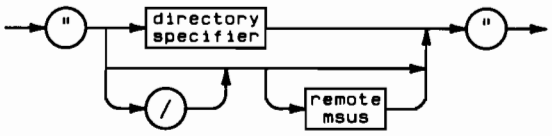
CAT PROTECT

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

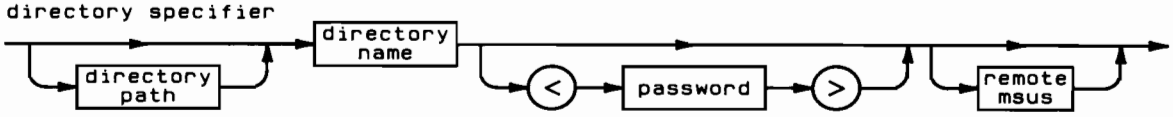
This statement lists all password-access capability assignments for a specified Shared Resource Management file or directory.



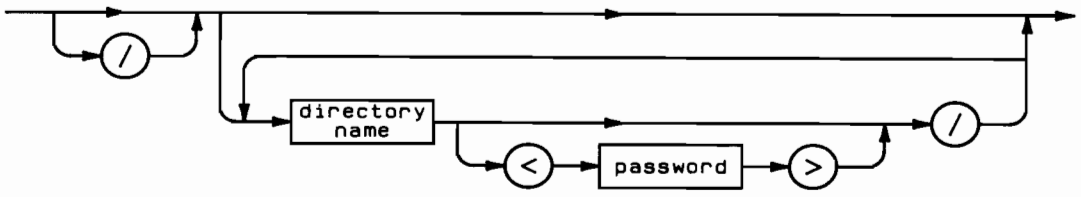
cat specifier



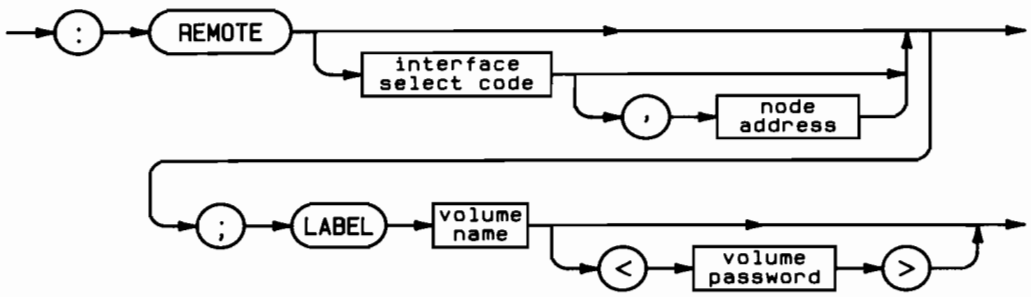
directory specifier



directory path



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices; remote: Integer constant assigned to the workstations 98029 interface. Default = 5.	(see Glossary)
HP-IB address	integer	0 thru 30
cat specifier	string expression	(see drawing)
heading suppression	simple numeric. 1 suppresses heading. Default = Heading printed	simple numeric
directory specifier	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)
password	string expression	any character except >
device type	letter specifying mass storage device	(see HP 9835/9845 BASIC manuals)
controller address	specifies a disc drive contoller	0 thru 7
unit code	integer designating disc unit; Default = 0	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

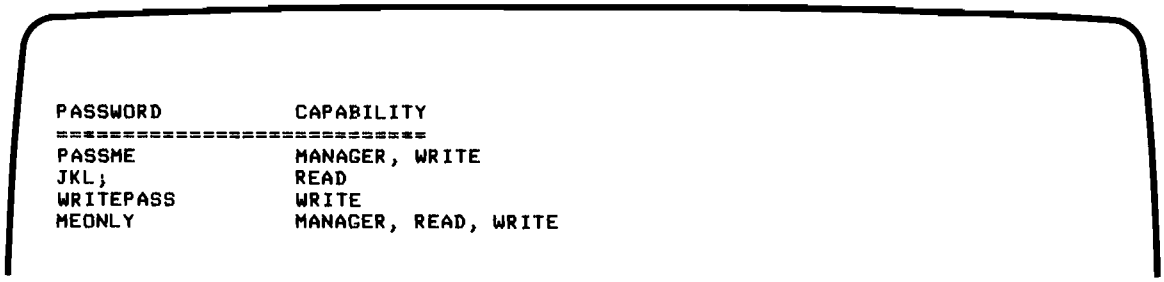
```
CAT PROTECT "/Dir1/Dir2/Test <Passtest>"
CAT PROTECT
CAT PROTECT "",0
```

Semantics

This statement lists all password-capability pairs for the file or directory specified. If you do not include an interface select code and HP-IB address in the statement, then the information will be sent to the current PRINTER IS device. Otherwise, it will be sent to whichever device corresponds to the select code and HP-IB address you specify.

If the heading suppression variable rounds to 1, the PASSWORD and CAPABILITY headings will not be printed. If control codes are part of the password, the display may be garbled. To remedy this problem, you need to turn on the workstation's Display functions mode. You do this with a PRINT " E_cY". When you have finished with the CAT PROTECT statement, turn off the display functions mode with a PRINT " E_cZ" statement.

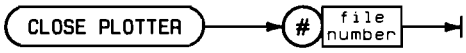
CAT PROTECT Display Example



CLOSE PLOTTER

Option ROMs Required	SRM, Graphics
Keyboard Executable	No
Programmable	Yes
In an IF...THEN...	Yes

This statement closes the file opened by PLOTTER IS #.



Item	Description/Default	Range Restrictions
file number	numeric expression	1 thru 10

Example Statements

```
CLOSE PLOTTER #
CLOSE PLOTTER #10
```

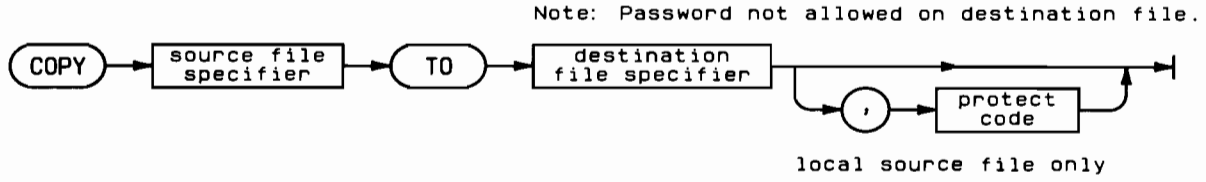
Semantics

The CLOSE PLOTTER # statement closes the file identified by the file number. This number may be 1 thru 10. If the file is in the spooler directory on the SRM System, the file is sent to the plotter. This statement is functionally similar to performing an ASSIGN #N TO * statement with a PLOTTER IS OFF statement.

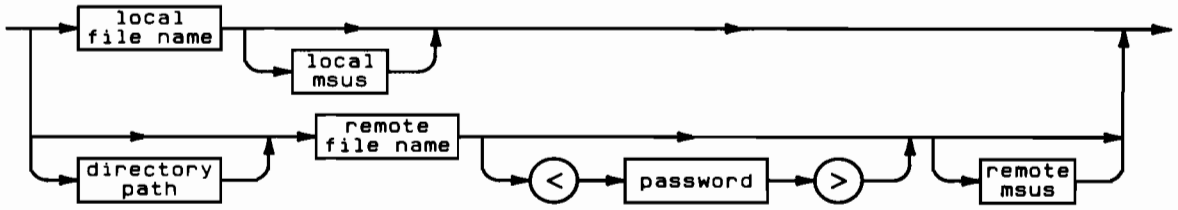
COPY

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

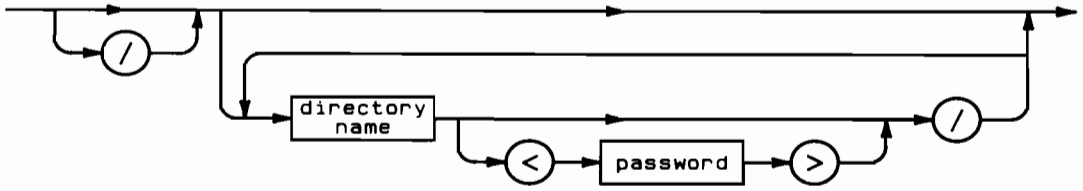
This statement creates a file and then copies the contents of a specified file into the newly created file.



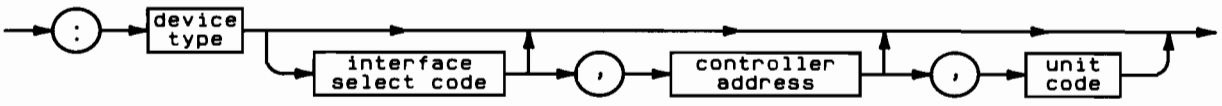
file specifier



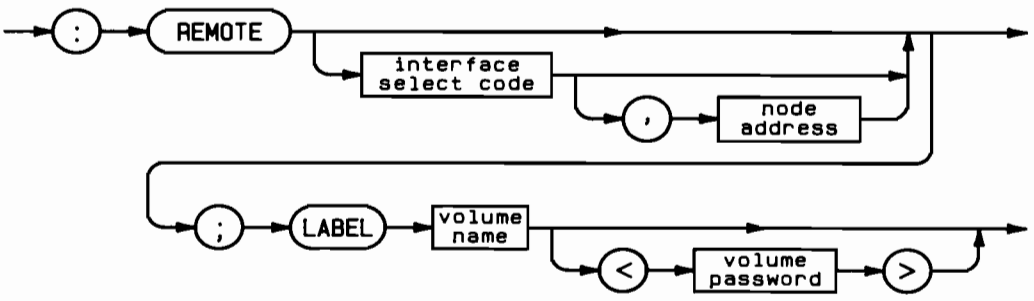
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
protect code	string expression. First 6 characters significant.	no null strings
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface; Default = 5	(see glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	volume name
string expression	(see glossary)	volume password
string expression	(see Glossary)	

Example Statements

```
COPY "/DirERIC/DirKAREN/Testfile <Testpass>" TO "/DirKRISTY/Newtest"
COPY "Newprog<Progpass>" TO "Masterdir/9841Test:REMOTE;LABEL Vol1"
COPY "PROTA:T15" TO "DIR1/DIR2/FILEONE:REMOTE"
```

Semantics

Files can be copied from:

- Workstation local mass storage to shared resource mass storage
- Shared resource mass storage to workstation local mass storage
- Shared resource volume to same shared resource volume
- Shared resource volume to a different shared resoure volume
- Share resource system to a different shared resource system

Any file types can be copied to the shared resource system. HP 9835 and HP 9845 files can be copied to either HP 9835 or HP 9845 local devices, but they are not executable except on their respective computer. The system checks to see that the destination file specifier doesn't currently exist, then, either gives you an error message telling you to select another name, or creates a destination file. Then all contents of the source file are copied into the new file. To copy an entire media, each file must be individually copied.

Files that have protected READ capability cannot be copied unless the correct password is given. The passwords themselves are not copied. If you want to restrict access to files you have just copied, you will have to protect them using the PROTECT statement. READ and WRITE capabilities are required in the immediately superior directory of the destination file for this statement to execute. DATA type files that have been initially SAVEd or CREATEd on a shared resource system do not have End-Of-File (EOF) marks. A pointer is used to mark the end of the file. When a file of this structure is copied to a local media, a potential problem exists if you use random access on the file. If you attempt to read from file records that have not been written, the system may not give you an error, but the results will be unpredictable. To avoid this potential problem, initialize all empty records in the file before it is copied from a remote to a local media.

Examples

```
COPY "FILE1:T15" TO "FILE1:REMOTE 5,0","PASS" EXECUTE
```

Copies FILE1 on local tape cartridge (unit T15) to a shared peripheral disc. The local file was protected by the protect code "PASS".

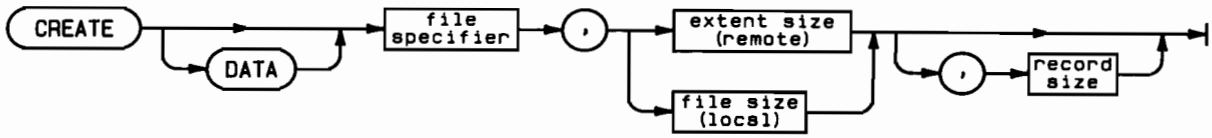
```
COPY "DIR1/TEST:REMOTE 5,0;LABEL Vol1" TO "TEST:T15" EXECUTE
```

Copies the file TEST from a shared resource volume named VOL1 to local tape unit T15. Note that we can mix local and remote msus's in the same statement.

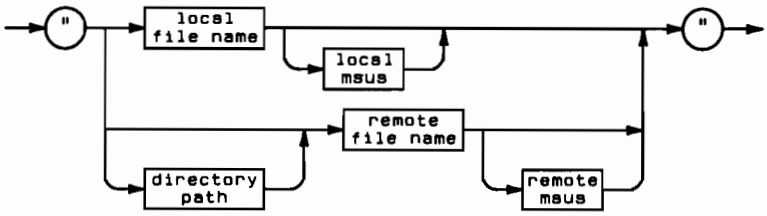
CREATE

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

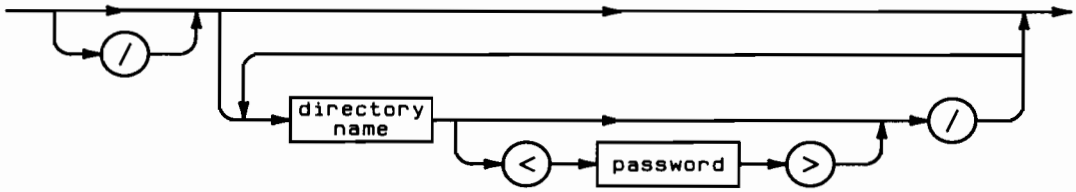
This statement is used to create a DATA file in the specified directory.



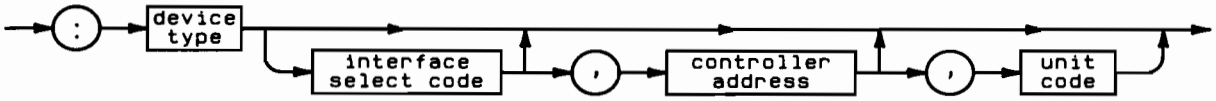
file specifier used with create statements



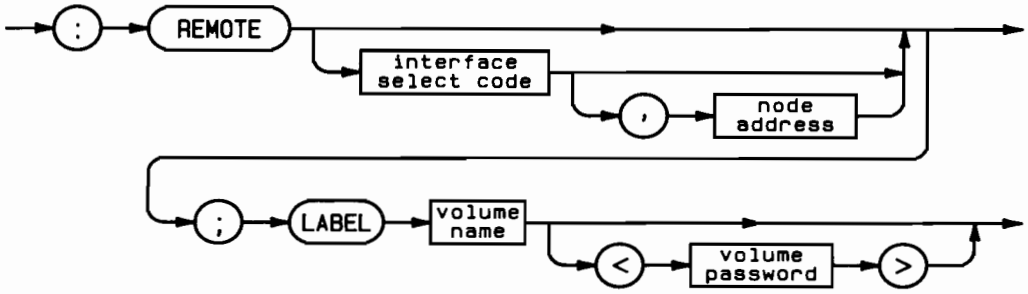
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
extent size	simple numeric	1 thru 32 767
file size	total file size in bytes	
record size	numeric rounded to an even number. Default = 256	4 thru 32 766 bytes
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)
password	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9835/9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface; Default = 5	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
CREATE DATA "/DirERIC/DirKATHY/Projectfile" ,100
CREATE "Afile", 10, 128
CREATE F$,50
A$ = "myfile"
B$ = ":REMOTE"
CREATE A$&B$,100
```

Semantics

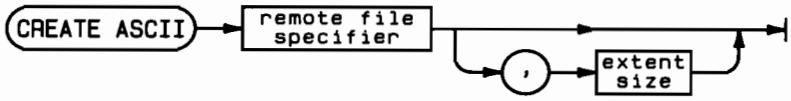
This statement creates an HP 9835 or HP 9845 DATA file. This file is compatible between the HP 9835 and HP 9845, but is not usable by the HP 9826 or HP 9836 workstations. The file stores numerics in Binary Coded Decimal format with each data item having a header. These headers are a form of type checking when entering data from the file. The keyword DATA is optional. The record length defines the number of bytes in each record. For remote access; the extent size sets the number of records for each extent. Whenever data is written to the file past the current extent size, the file is automatically extended by the number of records specified in the CREATE statement.

You need READ and WRITE capabilities for the directory that you are creating the file in for this statement to execute.

CREATE ASCII

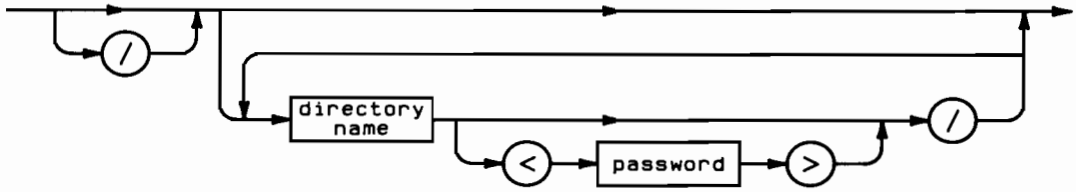
Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement is used to create an ASCII file on the shared resource system.

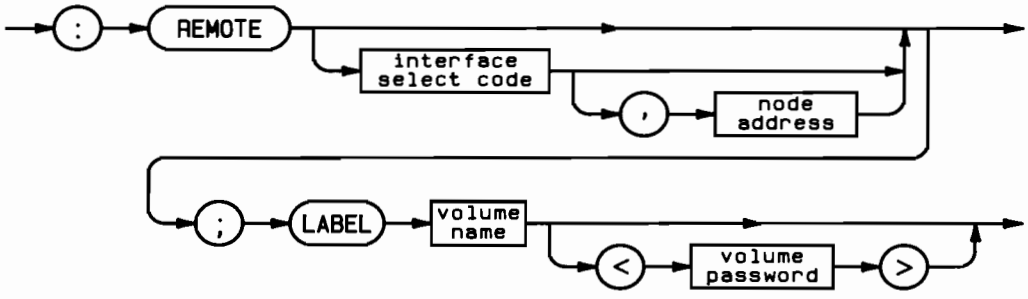


remote file specifier used with CREATE ASCII statement

directory path



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
extent size	simple numeric 1 thru 32 767; Default = 10 records	
directory path	string expression	(see drawing)
file name	string expression	(see Glossary)
password	string expression	(see Glossary)
msus	string expression	(see drawing)
directory name	string expression	(see Glossary)
interface select code	integer constant assigned to the workstations 98029 interface. Default = 5.	(see Glossary)
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation. Default = 0.	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
CREATE ASCII "/DirERIC/DirKathy<passme>/Projectfile", 100
CREATE ASCII "Afile:REMOTE",25

A$ = "newasciifile"
B$ = ":REMOTE"
CREATE ASCII A$&B$
```

Semantics

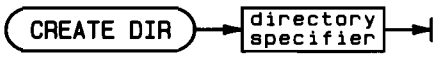
CREATE ASCII creates a new ASCII file by placing an entry in the specified (or default) directory. The HP 9835/9845 workstation can only use this command when in remote. This statement does not open the file when it creates it. ASCII files do not permit random access. They also do not have the data type checking performed with DATA files. Each data item in an ASCII file is a string of characters coded in the ASCII code. These files are extendable. That means that when the file is full, the system automatically adds another section of disc space equal to the extent size parameter defined with this statement. Record size is always 256 bytes/record. You must have READ and WRITE capabilities on the directory in order to create an ASCII file in that directory.

ASCII files are the only file type that is completely interchangeable with all other SRM workstations.

CREATE DIR

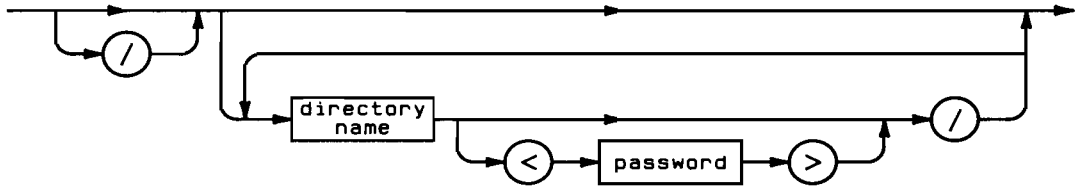
Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement creates a directory file within the shared resource management system.

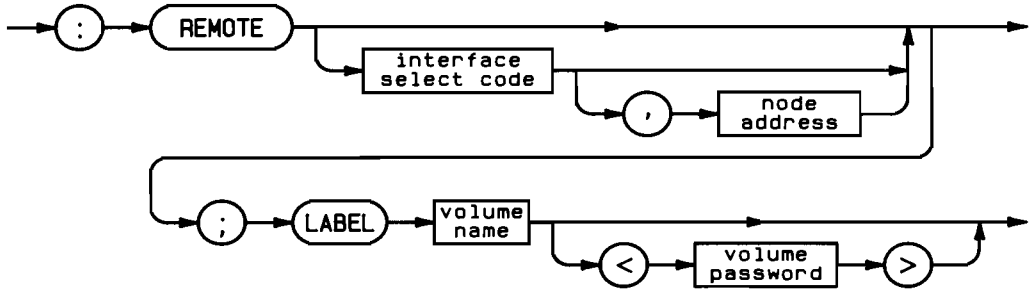


directory specifier for CREATE DIR statement.

directory path



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
directory specifier	string expression	(see Glossary)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)
msus	string expression	(see drawing)1
interface select code	integer constant assigned to the workstation's 98029 interface. Default = 5	(see Glossary)
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
CREATE DIR "upgrade_project:REMOTE"

MSI ":REMOTE"
CREATE DIR "DIR1/DIR2/NEWDIR"
```

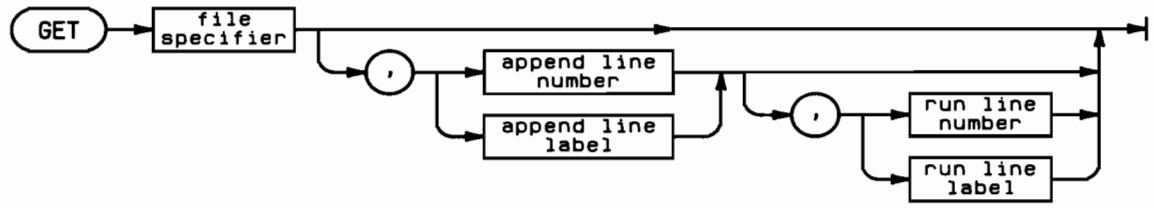
Semantics

The CREATE DIR statement allows the user to create a directory file within the shared resource hierarchical directory structure. This file is a fixed-format file and therefore allocation of memory space is automatically handled by the system. The new directory is created as a subordinate file to the directory specified in the directory specifier parameter of the CREATE DIR statement. Only one directory can be created at a time. All directories in the directory specifier parameter must be correctly specified with appropriate passwords in order for this statement to execute. You must have WRITE capability in the immediately superior directory to the directory you are creating.

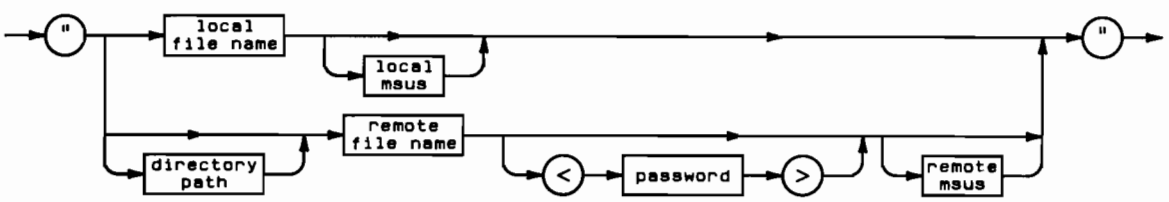
GET

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

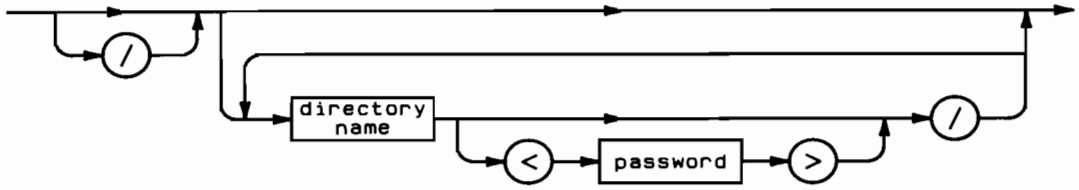
This statement places in internal memory a program saved in a DATA or ASCII file.



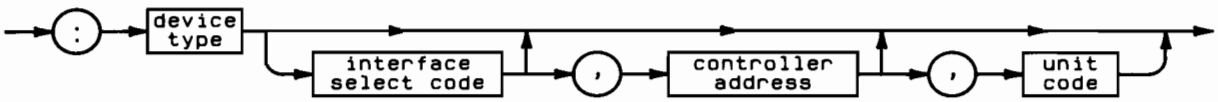
file specifier



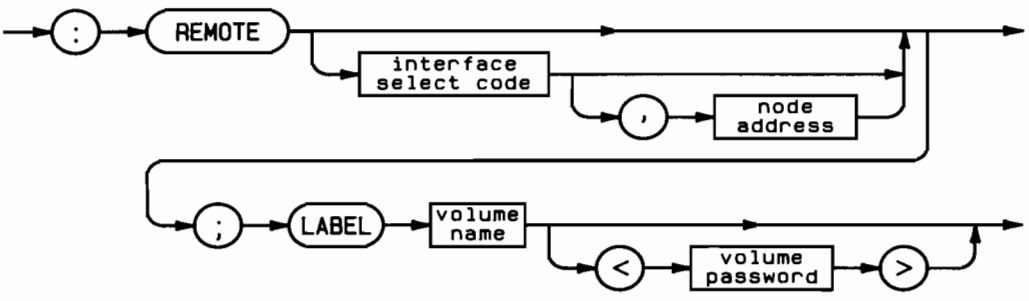
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
append line number	integer constant identifying a program line	1 thru 32 766
append line label	line label	any valid label
run line number	integer constant identifying a program line	1 thru 32 766
run line label	line label	any valid label
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
GET "/DirERIC/DirKATHY/Projectfile <Kathyonlypass>"
GET "Afile", 10, 120
```

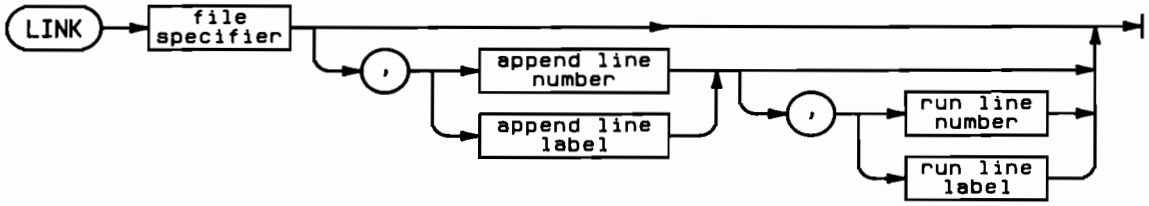
Semantics

This statement is the partner to the SAVE statement. GET retrieves and puts into memory a DATA file saved previously with the save statement or an ASCII file saved previously with the SAVE ASCII statement, or any string data file consisting of valid BASIC statements preceded by line number, stored one line per string. Execution of a GET statement cancels any tracing modes in effect. You need READ access to execute this statement.

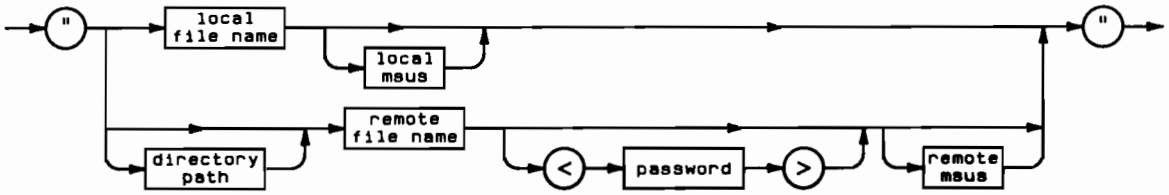
LINK

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

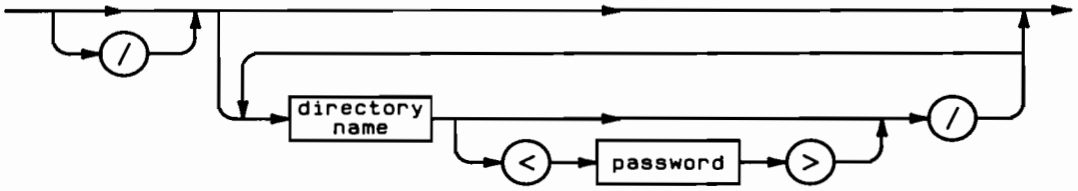
This statement places in internal memory a program saved in a DATA or ASCII file, including all values stored as variables.



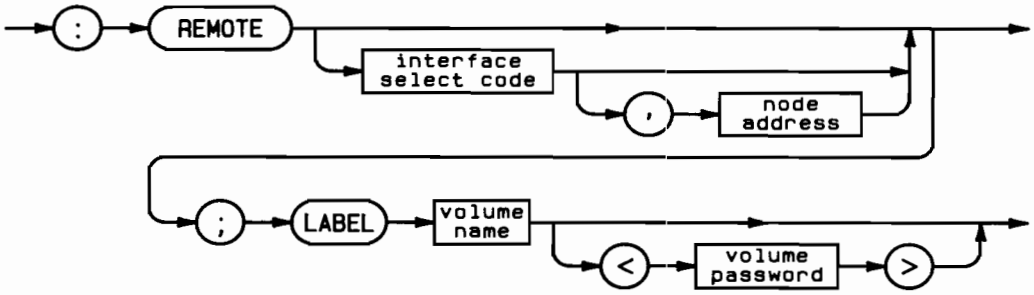
file specifier



directory path



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
append line number	integer constant identifying a program line	1 thru 32 766
append line label	line label	any valid label
run line number	integer constant identifying a program line	1 thru 32 766
run line label	line label	any valid label
file name	string expression (see Glossary) msus string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

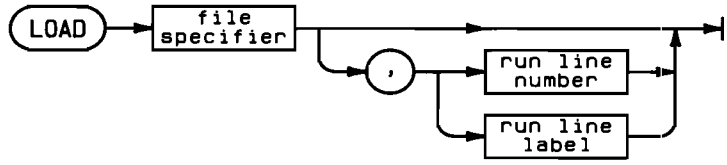
```
LINK "/DirERIC/DirKATHY/Projectfile <Kathyonlypass>"
LINK "Afile", 10, 120
```

Semantics

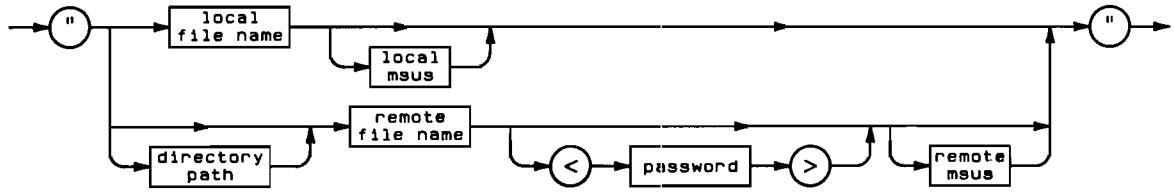
The syntax and operation of LINK is exactly the same as get except that LINK retains all values stored in variables, while GET only retains the values of COM variables. You must have READ capability to use this statement.

LOAD

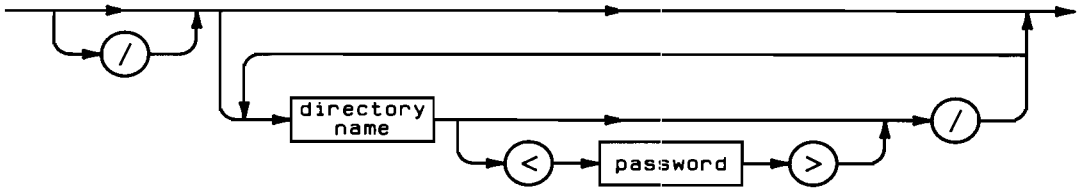
Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes



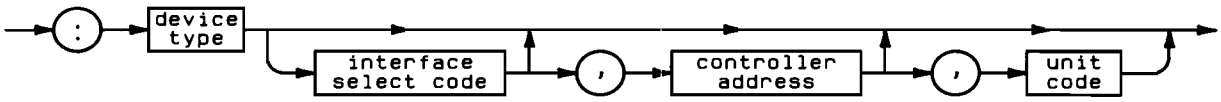
file specifier



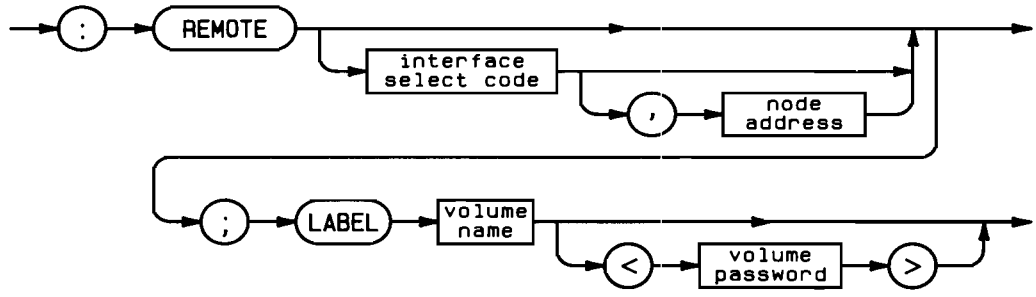
directory path



local mass storage unit specifier



remote mass storage unit specifier



This statement is the partner to the STORE statement. LOAD retrieves and puts into memory a PROG file stored previously with the STORE statement.

Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
run line number	integer constant	1 thru 32 766
run line label	string expression	valid line label
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
aunit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
LOAD "/DirERIC/DirKATHY/Projectfile <Kathyonlypass>"
LOAD "Afile" , 100
```

Semantics

Any BASIC program, and all variables not common are lost when LOAD is executed. It the COM area of the newly-loaded program does not match the existent COM area, the values in the old COM area are lost.

LOAD is allowed from the keyboard if a program is not running. If no run line is specified, RUN must be pressed to begin program execution. If a run line is specified. prerun initialization is performed and program execution begins at the specified line. The specified line must be in the main program context of the newly-loaded program. You must have READ capability to execute this statement.

Executing LOAD from a program causes a prerun, and program execution begins at either the specified run line or the lowest numbered program line in memory. If a run line is specified, it must be in the main program context of the newly-loaded program.

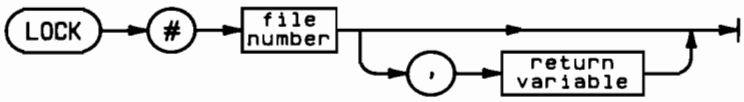
Program files created locally must be re-compiled before they can be used on the shared resource system. To do this, you first LOAD the program. Then you SAVE the program. The saving process converts the program into a DATA file format. Next you GET the file. And finally, you can STORE the file on the shared resource system (:REMOTE). For example:

```
LOAD "PROG1:T15" [EXECUTE]
SAVE "TEMP:T15" [EXECUTE]
GET "TEMP:T1" [EXECUTE]
STORE "PROG1:REMOTE" [EXECUTE]
```

LOCK

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement is the partner of the UNLOCK statement in providing sole access to a specified file.



Item	Description/Default	Range Restrictions
file number	integer ASSIGNed to a file	1 thru 10
return variable	numeric variable Lock successful: 0 Lock unsuccessful: 1	1 and 0

Example Statements

```
LOCK #7,Return1
LOCK #1
```

Semantics

Use the ASSIGN statement to open a file. The file will be opened in shared mode. To gain sole access to that file, use the LOCK statement. When you have finished the critical operation that required the LOCK (sole access), use the UNLOCK statement.

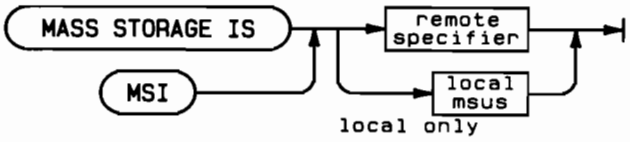
The return variable tells you if the lock was accomplished. If the file is already locked by another user, the return variable will show your LOCK statement failed. If you do not use the return variable, the system will give you an error message if the LOCK was not successful. Once locked, a file can only be unlocked by the same workstation that executed the LOCK statement.

LOCKs are cumulative. That is, if you lock a file more than once, you must either unlock it the same number of times, or close it. Closing a file automatically unlocks all locks. The reason for this can be seen from the following description. Let's say that you lock a file and then call a subprogram which you also lock. You then UNLOCK the subprogram. If the locks were not cumulative, the main file would be unlocked with the subprogram. This is not what was intended, so LOCKs are cumulative.

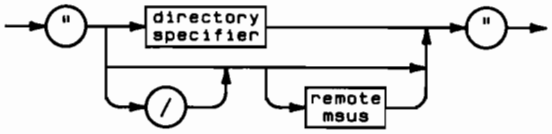
MASS STORAGE IS (MSI)

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

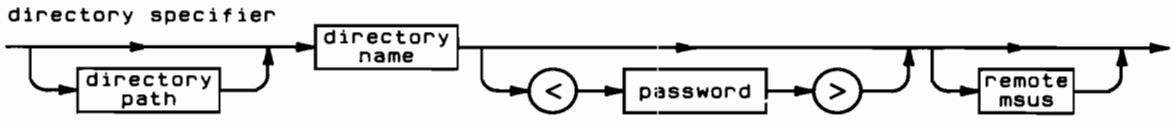
This statement specifies the system mass storage device and/or the current system working directory.



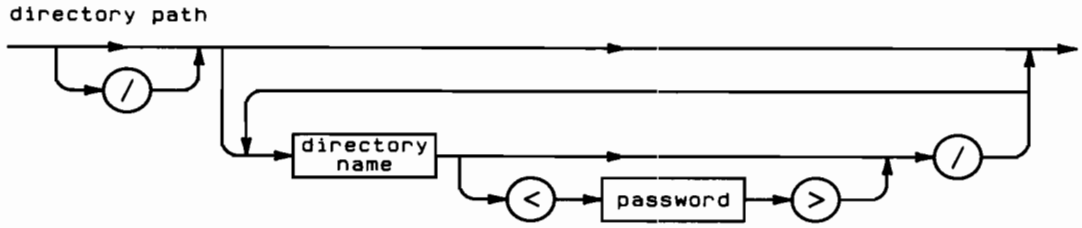
remote specifier



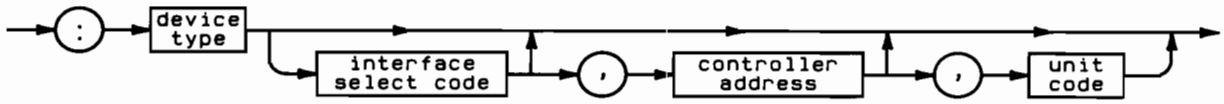
directory specifier



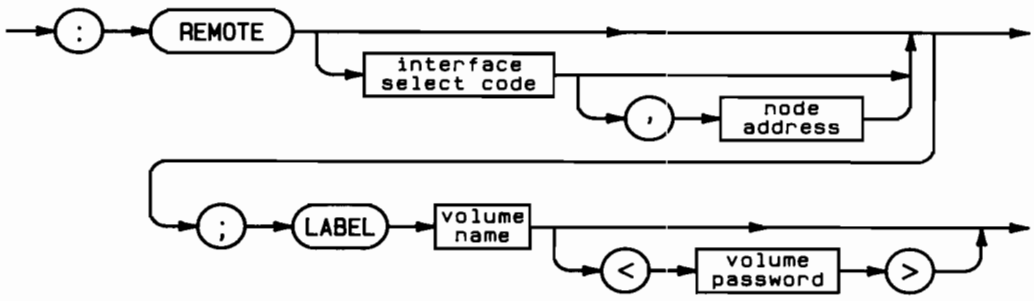
directory path



local msus



remote msus



Item	Description/Default	Range Restrictions
msi specifier	string expression	(see drawing)
msus	string expression	(see drawing)
directory specifier	string expression	(see drawing)
directory specifier	string expression	(see drawing)
directory path	string expression	(see Glossary)
password	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant remote: assigned to the workstation's 1 thru 12 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation Default = 0.	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

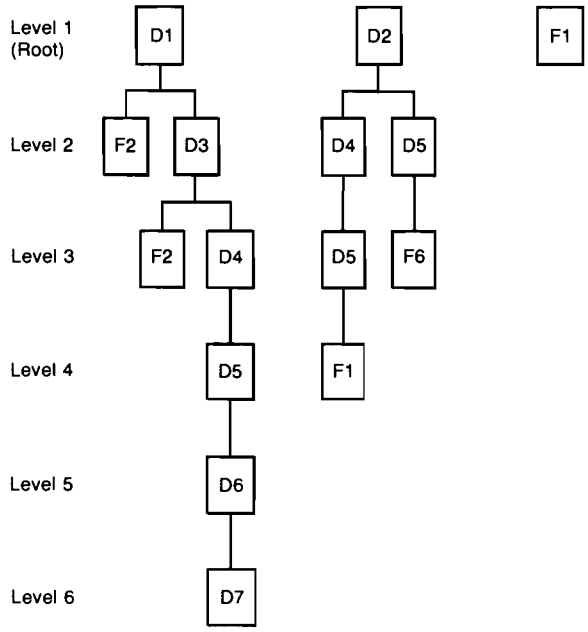
Example Statements

```
MSI ":REMOTE"
MSI "/Dir1/Dir2/DirDERALD<Meonly>"
MSI ".<password>"
MSI "A"
MSI ".."
MSI "/"
MSI A$
```

Semantics

Given the following directory structure, the following MSI statements have the indicated effect:

Directory Structure



Several directories and files have the same name. As long as the path from the root directory to the destination file or directory is unique, the same name can be used many times.

```
MSI ":REMOTE"  EXECUTE
```

Working directory is now the root directory. You have just entered the shared resource system. MSI ":T15" or MSI ":F8" will return you to local operation.

```
MSI "Dir1"  EXECUTE
```

If this statement follows the first (REMOTE) statement above, the directory Dir1 becomes the current working directory. You may now ASSIGN any file located in directory Dir1, say File2.

```
MSI "/Dir2/Dir4/Dir5"  EXECUTE
```

This statement begins with the root directory and moves through Dir2 and Dir4 to Dir5 which is now the working directory. The leading slash (/) tells the system to begin the path with the root directory. You must have previously "entered" the shared resource system for the slash to work. If you have not entered the system, the statement above must end with :REMOTE. If the leading slash is not there, then the current working directory is the default starting point.

To move back to directory Dir4 in the above example, type in:

```
MSI ".."  EXECUTE
```

The double periods indicates "move one directory closer to the root directory from the current working directory".

Let's say that directory Dir4 had its MANAGER capability protected with the password PASSME. We forgot to include the password, so we do not have the capabilty we need. A quick way to fix this, without re-specifying the whole MSI statement is to type in:

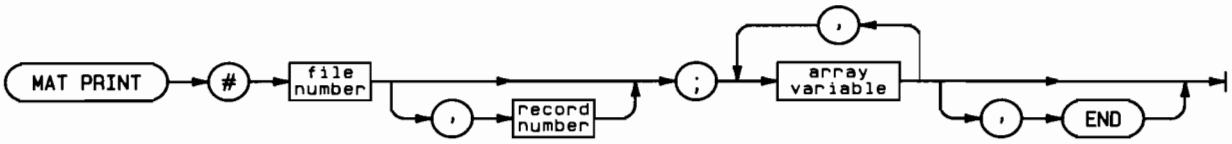
```
MSI ".<PASSME>"  EXECUTE
```

The single period means that you want the current working directory re-specified. The password is now specified and you now have MANAGER capability with the directory Dir4.

MAT PRINT

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN	Yes

This statement outputs data contained in array variables to DATA and ASCII files.



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
record number	integer	1 thru 32 767
array variable		see 9845 BASIC manuals

Example Statements

```
10 MAT PRINT #2;A,END
20 MAT PRINT #10,5;Real_array,Short_array,Integer_array,String_array$
30 MAT PRINT #File_number,Record_number;Another_array
```

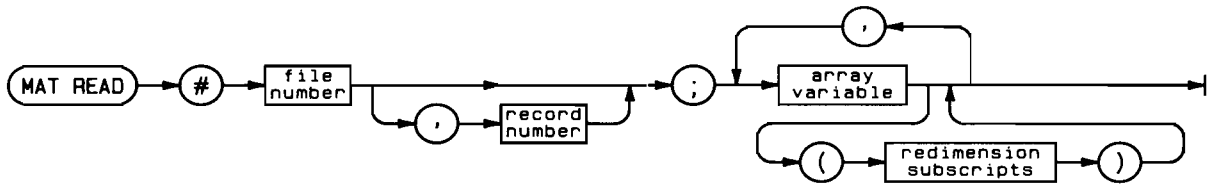
Semantics

MAT PRINT # performs the same function to arrays as PRINT # performs for other data. See the PRINT # statement for further information. You must have WRITE capability to execute this statement.

MAT READ

Keyboard Executable	Yes
Programmable	Yes
In an IF ... THEN	Yes

This statement enters data from DATA or ASCII files into arrays.



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
record number	integer	1 thru 32 767
array variable		(see HP 9845 BASIC manuals)
redimension subscripts	integers	(see HP 9845 BASIC manuals)

Example Statements

```
10 MAT READ #7;A,B(10,5)
20 MAT READ #3,32767;Read_array,Short_array,Integer_array,String_array$
30 MAT READ #File_number,Record_number;Another_array(M,N)
```

Semantics

MAT READ # performs the same function for arrays that READ # does for other data. See the READ # statement for further information. You must have READ capability to execute this statement.

OFF END

Keyboard Executable	No
Programmable	Yes
In an IF...THEN...	Yes

This statement cancels event-initiated branches previously enabled by an ON END statement.

Item	Description/Default	Range Restrictions
file number	integer ASSIGNed when file opened	1 thru 10

Example Statements

OFF END #4

Semantics

An ON END # declarative is deactivated with the OFF END # statement. OFF END # also reactivates OVERLAP mode for the file if it had been in effect previously.

OFF ERROR

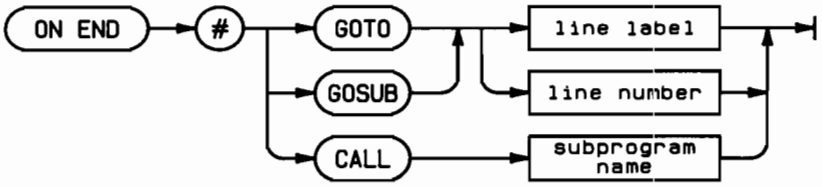
Keyboard Executable	No
Programmable	Yes
In an IF...THEN...	Yes

This statement cancels event-initiated branches previously enabled by an ON ERROR statement. Further errors are reported in the usual fashion.

ON END

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement is a declarative which causes a branching operation to occur when an EOF is encountered.



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
line number	integer numeric constant identifying a program line	1 thru 32 766 (9845) 1 thru 9999 (9835)
line label	name of a program label	any valid line label
subprogram	name of a subprogram defined by a SUB name statement	any valid name

Example Statements

```
ON END #1 GOTO 1234
ON END #10 GOSUB 4321
ON END #3 CALL Sub
```

Semantics

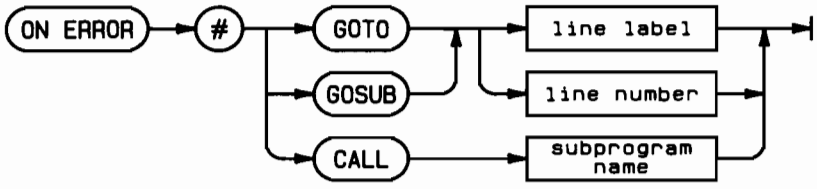
The ON END # statement is executed whenever an end-of-file (EOF) condition is encountered. You can have as many ON END # statements in a program as you like. They can be for different file numbers or for the same file number. When the system encounters an ON END # statement in a program, it will make a note of it and then proceed executing the program until an EOF is encountered. As soon as this occurs, the system will execute the command specified in the ON END statement. If you have more than one ON END statement in a program, the system will execute the statement it recorded last.

Executing an ON END # statement cancels an OVERLAP for that file. To turn off an ON END # declaration, use the OFF END # statement.

ON ERROR

Keyboard Executable	No
Programmable	Yes
In an IF...THEN...	Yes

This statement defines and enables an event-initiated branch which results from a trappable error. This allows you to write your own error handling routines.



Item	Description/Default	Range Restrictions
line label	name of program line	any valid name
line number	integer constant identifying a program line	1 thru 32 767 (9845) 1 thru 9999 (9835)
subprogram name	name of a SUB program	any valid name

Example Statements

```
ON ERROR GOTO 1200
ON ERROR CALL Report
```

Semantics

Run-time errors are those which occur only when the program is running. Dividing by zero is an example. A run-time error normally halts execution. Through use of the ON ERROR statement, run-time errors can be caught when they occur and execution can continue with the specified line. The ON ERROR statement specifies a branching which takes place after an error occurs.

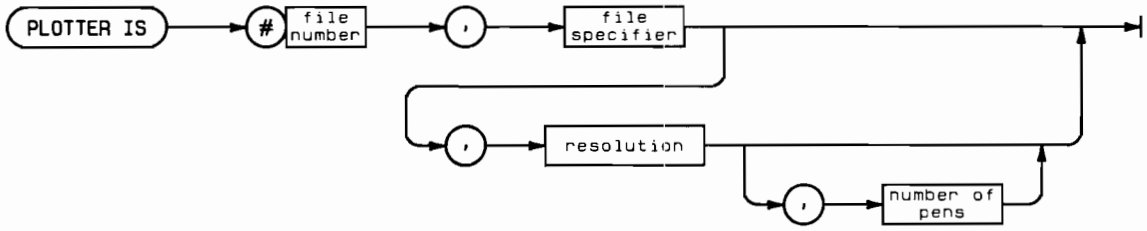
An ON ERROR...CALL statement is active in the program segment where it is declared and in all program segments called by that segment. ON ERROR...GOTO or GOSUB is active only in the program segment where it is declared. Execution of another ON ERROR statement cancels the previous one.

Errors which occur on LOAD, STORE, GET, SAVE and LINK statements are not trappable.

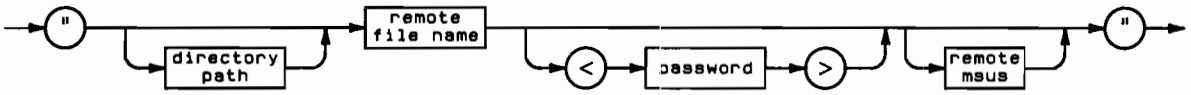
PLOTTER IS

Option ROMs Required	SRM, Graphics
Keyboard Executable	No
Programmable	Yes
In an IF...THEN	Yes

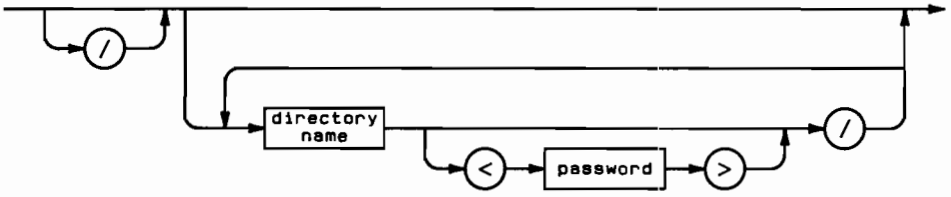
This statement associates a file number with an ASCII file. All subsequent plotting commands are sent to the file.



file specifier



directory path



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
file specifier	string expression	(see drawing)
resolution	numeric expression, resolution of graphics mapping; Default = .025 cm.	device dependent
number of pens	total number of pens accessible on multiple pen plotters, Default = 8	1 thru 8
directory path	string expression	(see drawing)
remote file name	string expression	any valid remote file name
password	string expression	(see Glossary)
remote msus	string expression	(see drawing)
directory name	string expression	(see Glossary)
interface select code	integer constant assigned to the workstation's 98029 interface. Default = 5	HP 9835 : 1 thru 14; HP 9845 : 1 thru 12
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
PLOTTER IS #File_number,File_specifier$
PLOTTER IS #4,File$&Msus$,0.05
PLOTTER IS #3,"FRED",0.025,6
```

Semantics

The PLOTTER IS # statement opens the file specified, associates the file number with the open file, and prepares the file for receiving plotter commands. This statement is functionally similar to performing a PLOTTER IS "HPGL" statement together with an ASSIGN # statement.

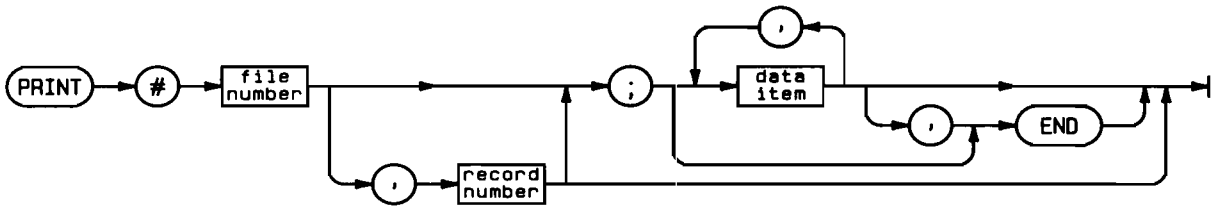
The plot file can only be an ASCII file. The use of another type of file produces indeterminate results.

Resolution is device dependent and specifies the incremental distance between points. The default value is 0.025 cm.

Number of pens specifies the total number of pens accessible on multiple pen plotters and is also device dependent. The default value is 8.

PRINT

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
record number	integer	1 thru 32 767
data list list of variables, array identifiers, number or strings of characters		

Example Statements

```
PRINT #2,3;A$,END
PRINT #3,2678;END
PRINT #1;A(*),B$,C$(*),3.1415926
```

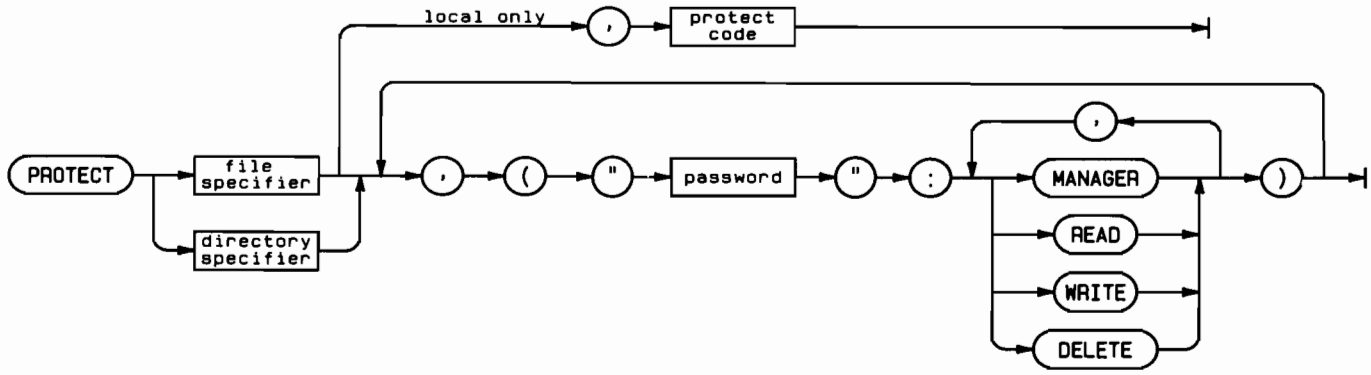
Semantics

This statement outputs data to DATA and ASCII files (and BDAT files if BDAT binary is installed). ASCII files permit sequential access only (defined record number cannot be used). ASCII files permit string data only, numeric data must be converted to strings before outputting. DATA files permit both string and numeric data. DATA files may use random or sequential access. END writes an end-of-file mark for both ASCI and DATA files.

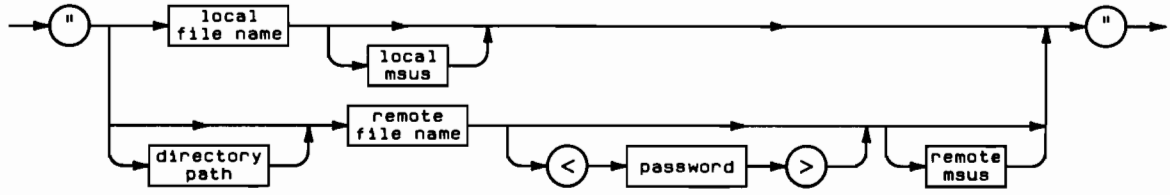
PROTECT

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

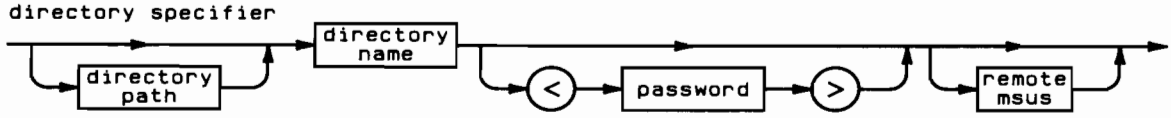
This statement assigns passwords and controls access capabilities for system directories and both local and system files.



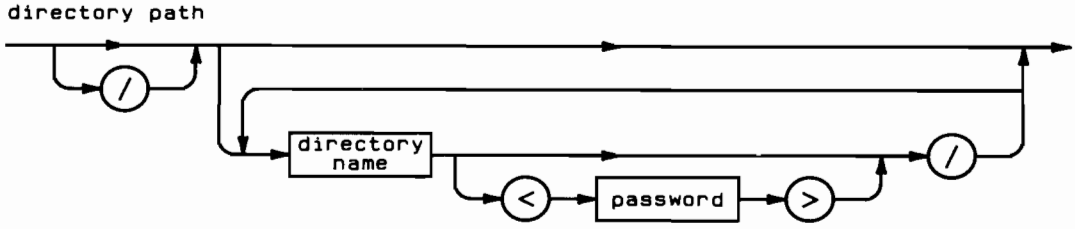
file specifier



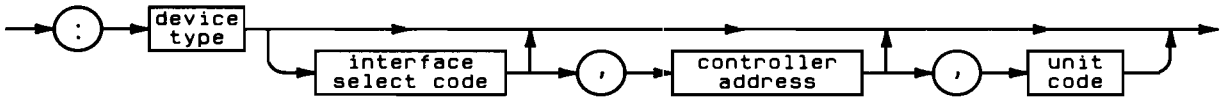
directory specifier



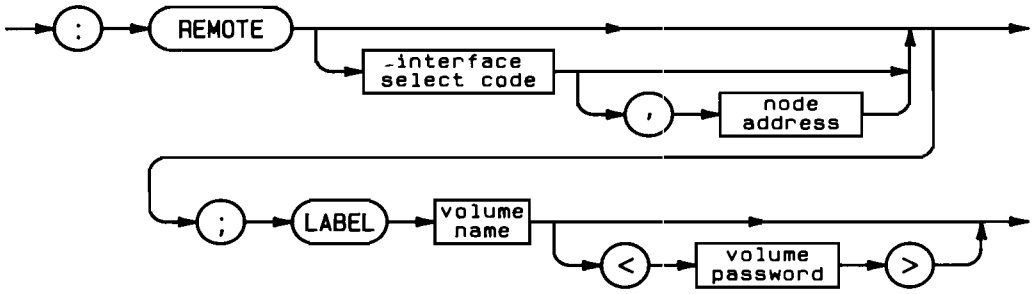
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
directory specifier	string expression	(see drawing)
protect code	string expression	see 9845 BASIC manuals
password	string expression	(see Glossary)
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant remote: assigned to the workstation's 1 thru 12 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation; Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
PROTECT "afile","secure"
PROTECT "Dir1/Dir2<passme>/Dir3",("password":MANAGER),("PASS":READ)
PROTECT "/Dir1/project_file",("fileprotect":WRITE)
```

Semantics

Local

This statement guards against accidental erasure.

Remote

This statement allows you to remove access capabilities from public access and PROTECT them with passwords. These capabilities are: MANAGER, READ and WRITE. The keyword DELETE is used to delete password assignments to a file or directory. Each PROTECT statement allows up to six password-capability combinations. However, there is no limit to the number of PROTECT statements you can issue per file or directory. Since PROTECT statements are cumulative, there is no limit to the number of passwords-capabilities combinations possible. The only rule is, each password must be unique. The only character not allowed for use in a password is the ">" character which indicates the end of the password.

To modify a password-capability pair, just re-protect the file with the same password and the new set of capabilities you want paired to that password. The new set must include all capabilities desired, including previously assigned capabilities. So to add or subtract capabilities to a password, just re-protect. Files must be closed to be protected.

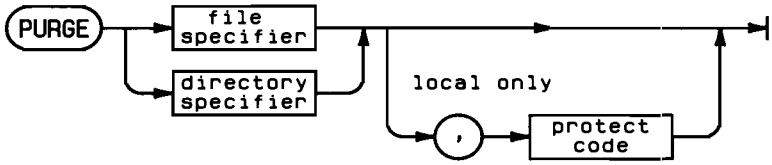
To delete a password, just use the PROTECT statement with the DELETE parameter.

You need MANAGER access to PROTECT a file. This means that if the file has previously been assigned a password for MANAGER capability, you must include this password in the file specifier when doing another PROTECT of that file or directory.

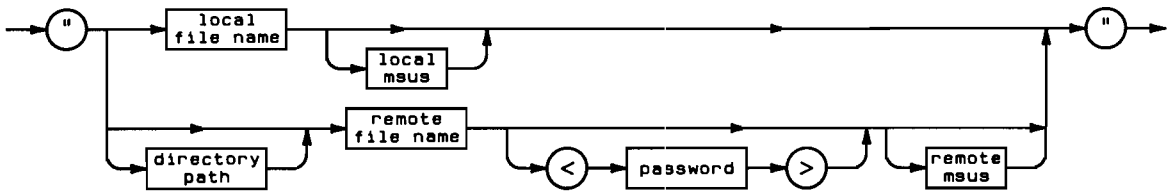
PURGE

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

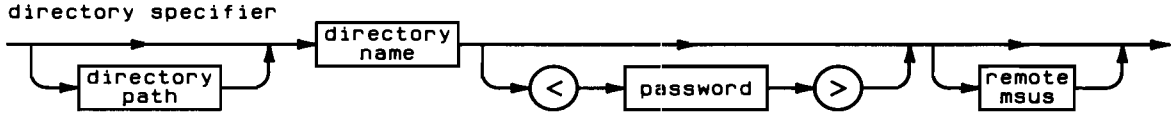
This statement deletes a file or directory entry from the specified directory on the specified mass storage volume.



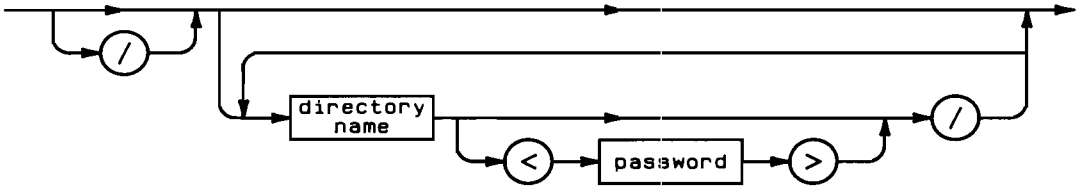
file specifier



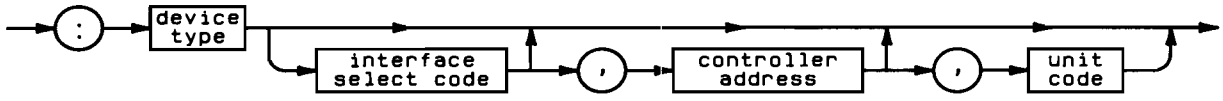
directory specifier



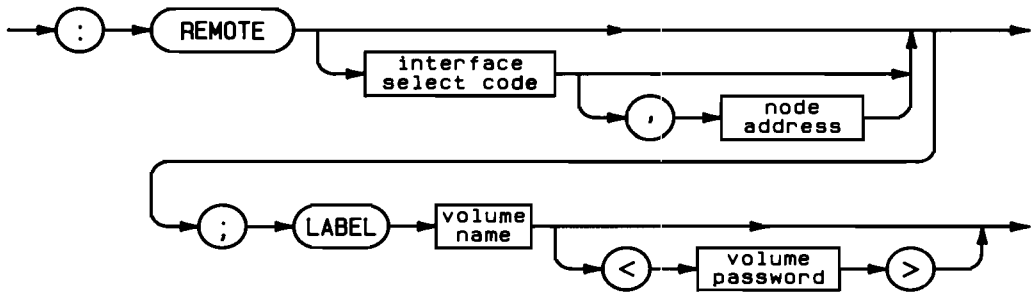
directory path



local msus



remote msus



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
directory specifier	string expression	(see drawing)
protect code	string expression. First 6 characters significant	no null strings
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
directory name	string expression	(see Glossary)
password	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9835/9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation Default = 0.	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
PURGE "afile:F8","X"
PURGE "/Dir1/Dir2<passme>/afile"
PURGE "/Dir1/Dir2"
```

Semantics

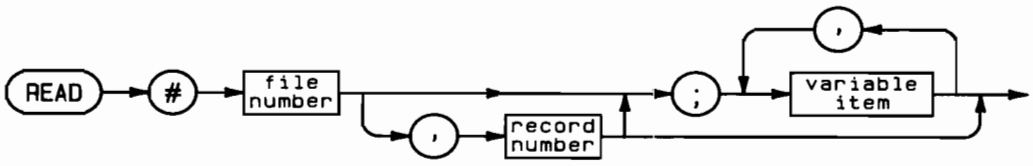
This statement removes a file or directory from the msus device. Removing a directory is possible only if there are no files nor directories below it in the hierarchical directory structure. The file or directory must also be closed before purging or an error will result. Appropriate protect codes and passwords must be included to remove a file or directory. When removed, the file's disc space is returned to the system.

You must have READ and WRITE capability in the directory immediately superior to the file or directory you wish to purge. MANAGER capabilty is also required for the file or directory you wish to purge.

READ

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement reads data items into a specified, open file.



Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10
record number	integer	1 thru 32 767
variable list	list of numeric and/or string variables separated by commas	any valid list of variable names

Example Statements

```
READ #4; A$, B$
READ #10, 12; D$, C$,A(*),X
```

Semantics

READ # takes data from the file specified and assigns it to the variable list. If the file is an ASCII type file, then the data can only be assigned to string variables. In addition, ASCII files only allow serial access. DATA type files, on the other hand, can hold both string and numeric data items and can be accessed both randomly and serially.

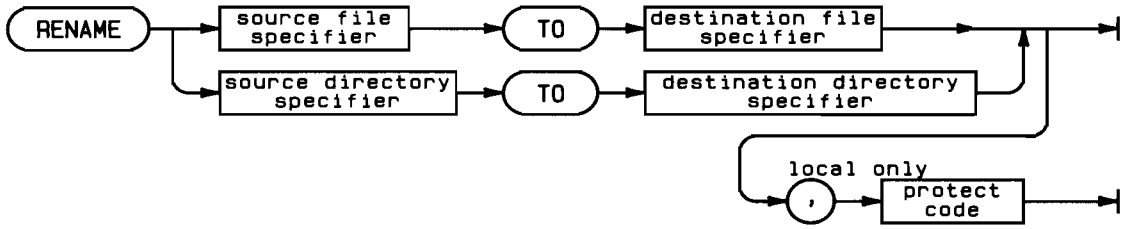
To access a file serially, specify the file number but not the record number. The system will read from record 1 of that file and enter each data item into a variable in the variable list in the order in which they occur. The variable type must agree with the data item type (i.e., strings must be assigned to string variables and numerics to numeric variables). When accessing a file serially, the system skips over EOR marks and goes on to the next record until all variables in the variable list have been filled. To READ files in random mode, specify a defined record number. The system will begin reading data at the beginning of the specified record. It will stop READing data as soon as it encounters an EOF pointer or EOR mark.

To reposition the data file pointer, specify a record number but not a variable list. The data pointer will move to the beginning of the specified record. To re-position the file pointer to the beginning of ASCII files, re-ASSIGN the file, or use READ #1,1. This is the only case where a defined record number can be used with ASCII files. If the file is locked by another user, the system will wait until unlocked for the READ # to complete.

RENAME

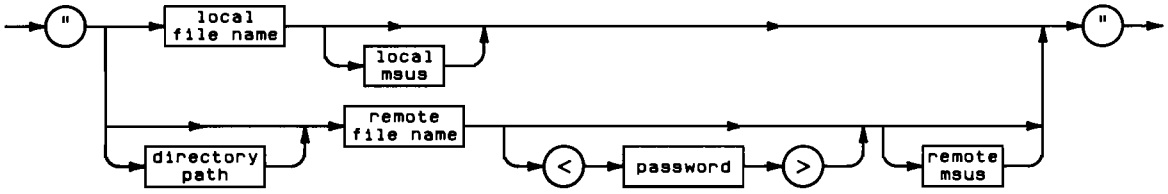
Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement is used to change the name of a specified file or system directory. File movement (relocation) is also supported for shared resource files and directories.

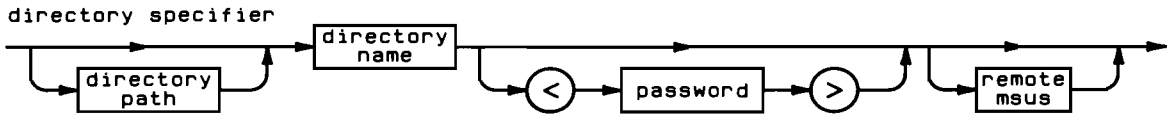


Note: Destination files and directories can not have mass storage unit specifiers nor passwords specified.

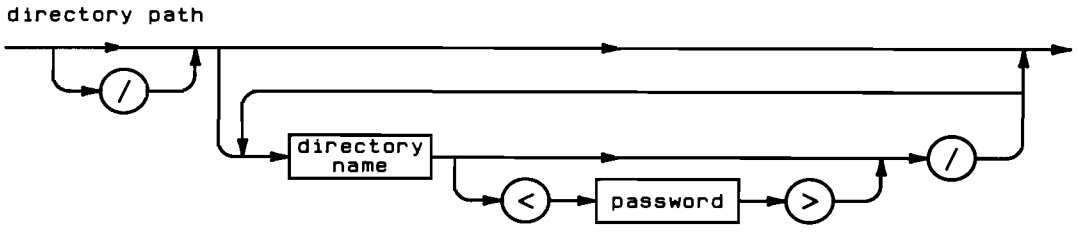
file specifier



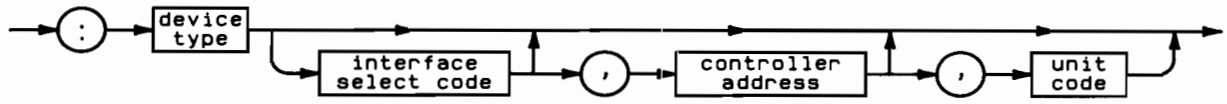
directory specifier



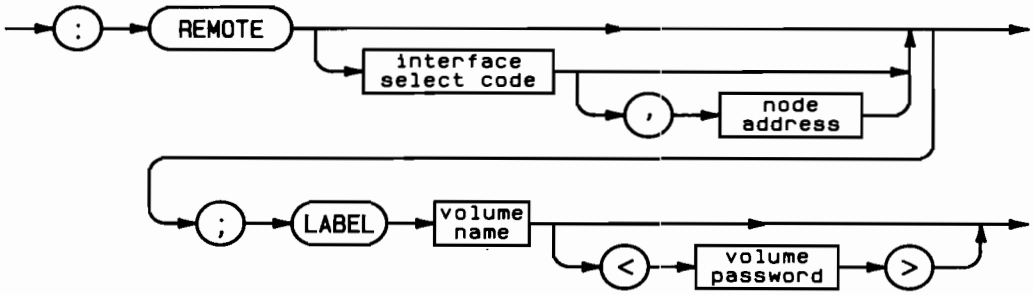
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
directory specifier	string expression	(see drawing)
protect code	string expression. First 6 characters significant	no null strings
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9835/9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation. Default = 0.	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
RENAME "afile" TO "new1"
RENAME "/Dir1/Dir2/afile<filepass>" TO "/Dir1/Dir3/Newfile"
RENAME "A:T15" to "B","pass"
```

Semantics

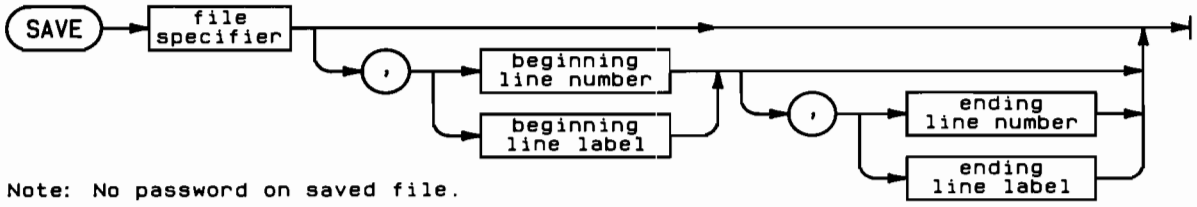
This statement renames a file or directory. In addition, files may be moved to other directories on the same volume in the shared resource system. The password-capability pairs are carried along with the new name. Files may not be moved across volume or device boundaries. Directories may not be moved. Files must be closed before renaming.

You must have READ and WRITE capability for the immediately superior directory to the file you wish to rename. You must have MANAGER capability for the file to rename it.

SAVE

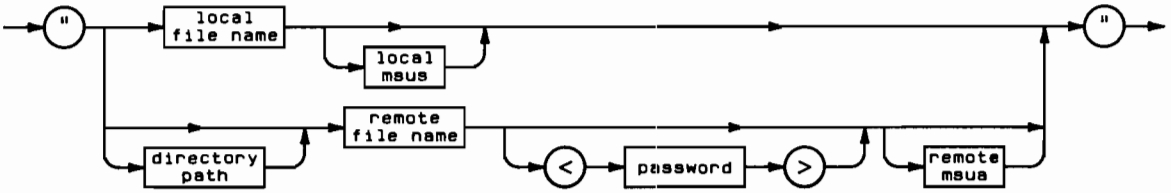
Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement saves the program in memory as a DATA file on the specified msus. SAVE is the partner of GET.

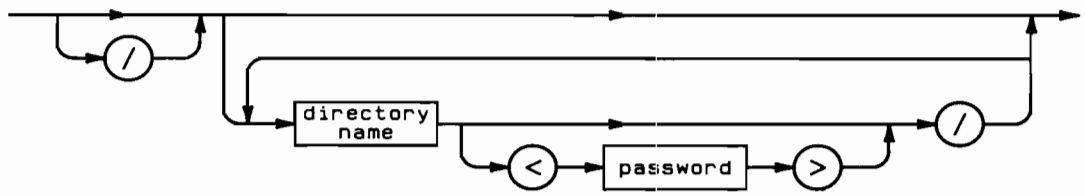


Note: No password on saved file.

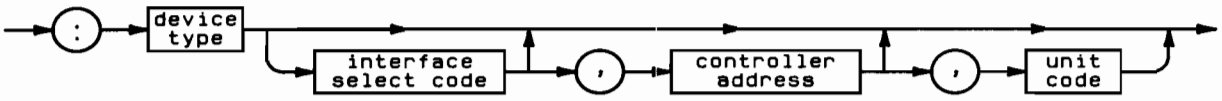
file specifier



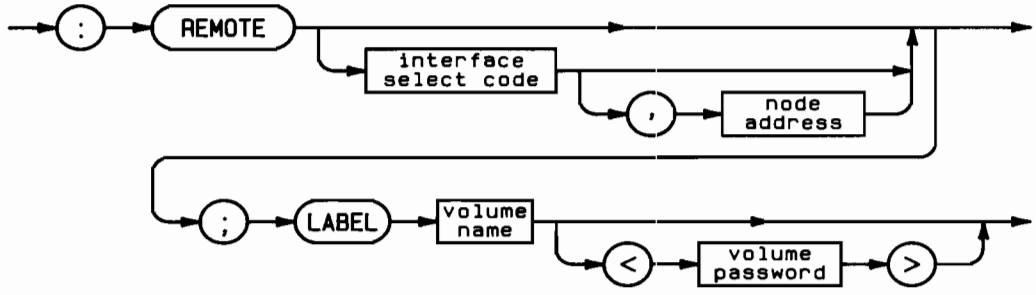
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
beginning line number	integer constant	1 thru 32 766
beginning line label	label	(see HP 9835/9845 BASIC manuals)
ending line number	integer constant	1 thru 32 766
ending line label	label	(see HP 9835/9845 BASIC manuals)
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9835/9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation. Default = 5.	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
SAVE  "life"
SAVE  "/Dir1/Dir2<passme>/Afile",100,200
```

Semantics

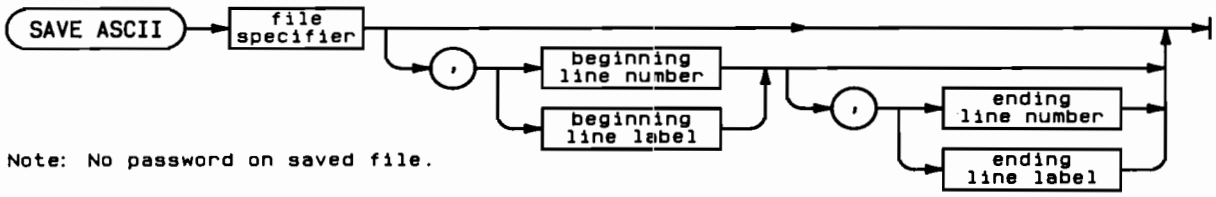
The SAVE statement creates a DATA type file and saves the program currently in memory in it. If you only want to save part of the program, specify a beginning line and/or ending line. Programs that have been SAVEd can be retreived with the GET command.

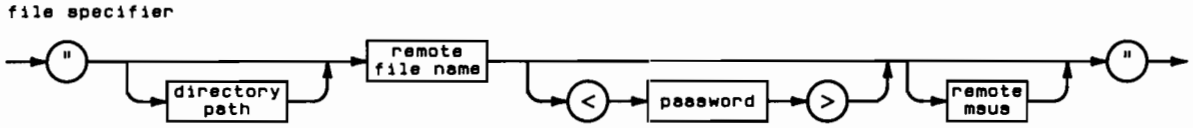
You must have READ and WRITE capabilities for the immediately superior directory to the file you want to save. You can also access these files with READ # and PRINT # if you want to manipulate the data.

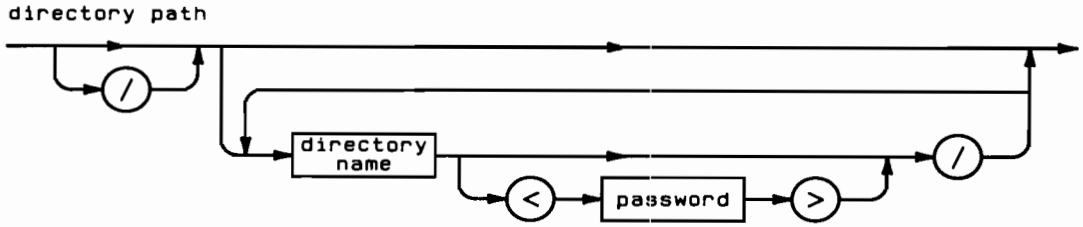
SAVE ASCII

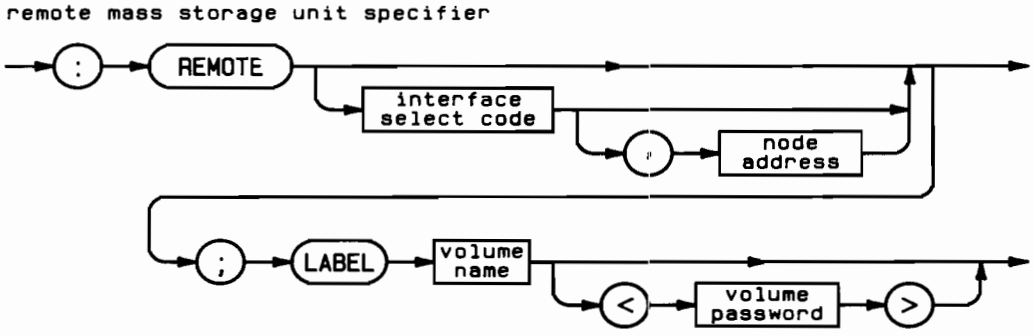
Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement saves ASCII files on the specified msus. SAVE ASCII is the partner of the GET statement.









Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
beginning line number	integer constant	1 thru 32 766
beginning line label	label	(see HP 9845 BASIC manuals)
ending line number	integer constant	1 thru 32 766
ending line label	label	(see HP 9835/9845 BASIC manuals)
directory path	string expression	(see drawing)
file name	string expression	(see Glossary)
password	string expression	(see Glossary)
msus	string expression	(see drawing)
directory name	string expression	(see Glossary)
interface select code	integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation. Default = 5.	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
SAVE ASCII "life"
SAVE ASCII "/Dir1/Dir2<passme>/Afile",100,200
```

Semantics

SAVE ASCII performs the same function for ASCII files that SAVE does for DATA files. This statement writes the program in the workstation's internal memory into the specified file in the form of ASCII strings.

You must have READ and WRITE capabilities for the immediately superior directory to the file you want to save.

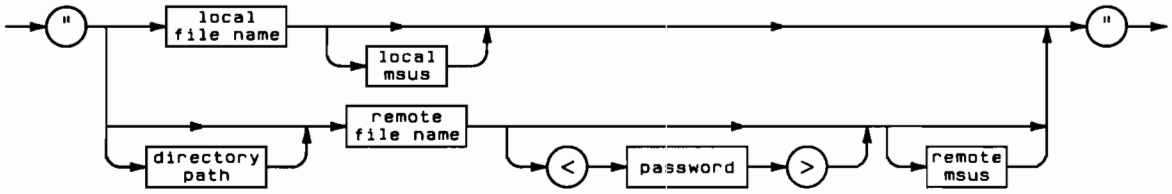
STORE

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

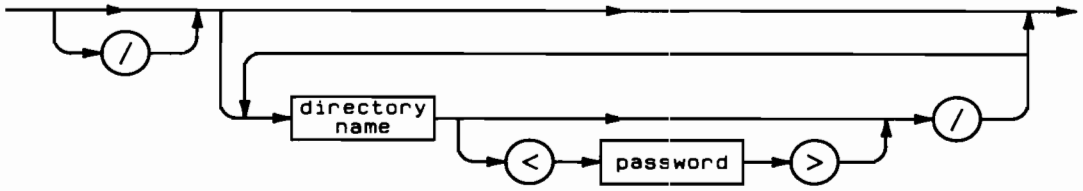
This statement stores all program lines and binary routines in memory into a program file on the specified mass storage device.

Note: No password on stored file.

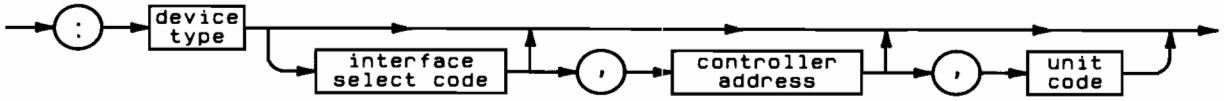
file specifier



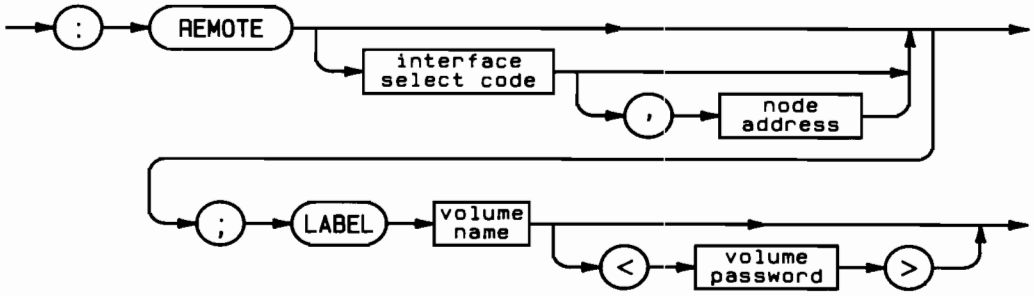
directory path



local mass storage unit specifier



remote mass storage unit specifier



Item	Description/Default	Range Restrictions
file specifier	string expression	(see drawing)
file name	string expression	(see Glossary)
msus	string expression	(see drawing)
directory path	string expression	(see drawing)
password	string expression	(see Glossary)
directory name	string expression	(see Glossary)
device type	letter specifying mass storage device	(see HP 9835/9845 BASIC manuals)
interface select code	local: integer constant with default of 15 for T devices, 8 for F devices, 7 for H devices and 12 for all other devices remote: integer constant assigned to the workstation's 98029 interface. Default = 5.	(see Glossary)
controller address	integer constant	0 thru 7
unit code	integer constant	0 thru 7
node address	integer constant assigned to controller's 98629 interface assigned to user's workstation. Default = 0	0 thru 63
volume name	string expression	(see Glossary)
volume password	string expression	(see Glossary)

Example Statements

```
STORE "file"
STORE "Dir1/Dir2/Afile:REMOTE"
```

Semantics

The STORE statement creates a PROG type file and stores the program and binaries currently in memory into it. Stored files are retrieved with the LOAD statement.

You must have READ and WRITE capabilities for the directory you want to STORE your file in.

UNLOCK

Keyboard Executable	Yes
Programmable	Yes
In an IF...THEN...	Yes

This statement works with LOCK to provide temporary sole access to a specified, open file.

Item	Description/Default	Range Restrictions
file number	numeric expression, rounded to an integer	1 thru 10

Example Statements

```
UNLOCK #2
UNLOCK #10
```

Semantics

Use this statement to UNLOCK an open file that you previously locked. You should UNLOCK a file as soon as you have completed the critical operation for which you locked it, so that other users may have access to the file. See the LOCK statement for further information.

Chapter 4
Converting HP 9835/9845 Programs to Use SRM

Mass Storage Conversions – File Name Definition

Some programs define a separate string for the file name and for the mass storage unit specifier, while others use one string for everything. Still other programs use MASS STORAGE IS statements throughout the program. The one-string programs are usually the easiest to convert to the SRM; the two-string programs are usually the next easiest; and the MASS STORAGE IS programs are usually the most difficult.

For most cases, simply increasing the size of a string is all that must be done to convert a one-string program. Since the maximum string size that can be entered from the keyboard is 160 characters, this is a good size for a file name string. The program will still default to the currently-defined mass storage device if the msus is not specified.

Example:

```
10    DIM Filename$[160]
      .
      .
      .
1000 LINPUT "Where would you like to store your data?",Filename$
1010 ASSIGN #1 TO Filename$
1020 PRINT #1;Data(*)
1030 ASSIGN #1 TO *
```

Note

On an SRM it is important to close files when they are no longer needed. If they are not closed in the program, they remain open after the program is finished.

On the other hand, suppose the program is structured like this:

Example:

```
10   DIM Filename$[6],Msus$[11]
     .
     .
     .
500  Filename$="SLIDES"
510  Msus$=":H8,0,1"
     .
     .
     .
1000 ASSIGN #1 TO Filename$&Msus$
1010 PRINT #1;Data(*)
1020 ASSIGN #1 TO *
```

In this case, it is probably easiest to add another string variable for the (optional) SRM directory path, for example:

Example:

```
10   DIM Pathname$[160],Filename$[34],Msus$[30]
     .
     .
     .
500  Pathname$="FRED/DATA_FILES/"
510  Filename$="SLIDES"
520  Msus$=":REMOTE 5,1"
     .
     .
     .
1000 ASSIGN #1 TO Pathname$&Filename$&Msus$
1010 PRINT #1;Data(*)
1020 ASSIGN #1 TO *
```

If the Pathname$ variable is null, the statement looks exactly like it did before the modification. If the Msus$ variable is null, the current mass storage device is accessed. The only difference is in the allowable length of the string variables. The file name variable has been extended to 34 characters so it will fit a 16-character file name, a 16-character password, and < > (e.g., "ABCDEFGHIJKLMNOP <abcdefghijklmnop>").

Programs that use MASS STORAGE IS statements with a literal (e.g., MASS STORAGE IS ":H8,0,1") should probably be converted either to using a variable in the MASS STORAGE IS statement, or to include the msus in the file name (the same structure as the one-string program type mentioned above).

Example:

```
10   DIM Filename$[6],Msus$[11]
     .
     .
     .
500  Filename$="SLIDES"
510  Msus$=":H8,0,1"
     .
     .
     .
1000 MASS STORAGE IS Msus$
1010 ASSIGN #1 TO Filename$
1020 PRINT #1;Data(*)
1030 ASSIGN #1 TO *
```

File Name Compatibility

Non-SRM and SRM file names have different constraints. The rules for non-SRM file names are:

- 1-6 characters;
- All characters are legal except the colon, quote mark, ASCII NULL (CHR$(0)), and CHR$(255);
- Blanks are ignored.

The rules for an SRM file name are:

- 1-16 characters;
- Legal characters are uppercase and lowercase letters, digit underscore (_), period, and "foreign" characters (CHR$(161) through CHR$(254));
- Blanks are ignored.

The set of file names that are legal on both non-SRM and SRM devices are:

- 1-6 characters;
- Legal characters are uppercase and lowercase letters, digits, underscore (_), period, and "foreign" characters (CHR$(161) through CHR$(254));
- Blanks are ignored.

Protect Codes

The protect code format for shared resource files is different from the format for non-SRM files, and a branch condition must be used. The alternatives for the branch are:

1. Try the non-SRM syntax with an ON ERROR statement enabled, and if an error occurs, see if it indicates that the mass storage device is an SRM. An ERROR 130 occurs when any of the following syntaxes with protect codes are attempted on an SRM:

   PROTECT file specifier, protect code

   ASSIGN #file number TO file specifier, return variable, protect code

   ASSIGN file specifier TO #file number, return variable, protect code

   PURGE file specifier, protect code

   COPY file specifier TO file specifier, protect code

   RENAME file specifier TO file specifier, protect code

   If any other error occurs, the appropriate action will have to be taken (e.g., printing out the ERRM$ and stopping the program).
2. If the program uses a string in which to store the mass storage unit specifier, check for the value of POS (msus,":REMOTE"). This alternative is easier to implement than alternative (1), but if the program allows the string to be empty in order to access the default device, this alternative will not work.

If the program looks for a protect code error (ERROR 62) at the time of an ASSIGN to see if a protect code is needed, the program will have to be modified: The SRM won't give the protect code error until a READ # or PRINT # is attempted, since either or both operations can be affected by the SRM protect code(s).

Other Needed Mass Storage Changes

The following statement types must be changed for the program to work on the SRM:

If TYP is used, another method of accomplishing the same thing must be used (e.g., an ON ERROR trap while trying to read different data types from the file).

If RE-SAVE (or RE-STORE) is used, it must be replaced with PURGE, then SAVE (or STORE), then PROTECT if it is needed. Or, to be safer (in case of a device failure), replace it with SAVE (or STORE) under a temporary name, then PURGE the old file, then RENAME the new file, then PROTECT if needed.

If LOAD KEY is used, it must be replaced with something else, such as an ON KEY or ON KBD routine.

REWIND msus must be conditional, because the SRM gives an ERROR 52 if a REWIND is attempted on it.

Example:

```
1000 IF POS(Msus$,":T") THEN REWIND Msus$
```

FCREATE, FPRINT, and FREAD statements must be replaced with normal CREATE, PRINT #, and READ # statements when using SRM.

CAT TO$ routines need to check for the SRM (e.g., reading the device type from the CAT header) since the data appears in different positions depending on whether the CAT is done on an SRM or a non-SRM device.

Reading the header of a CAT (using CAT TO $) can no longer be used to determine the current mass storage device. Only the current directory (the entire directory path) is printed in the SRM CAT header. One alternative is to query the program user for the current directory path if the device is an SRM, but there is no clean work-around.

ILOAD and ISTORE are not allowed with SRM. The only way to get around this is to store assembler files on a tape or other non-SRM medium, or to assemble the code in the program.

Printer Spooling

Printer spooling on the SRM is not a simple task of saying PRINTER IS ":REMOTE". Printer spooling entails creating a file in the spool directory and outputting strings to the file. After the file is closed, the SRM takes control of it, locks it, prints it out on the printer, and purges it.

Creating a Spooler File

The best type of name for a spool file is a short name descriptive of the data in the file and a number (e.g., "TEXT3", "Data_Points_42"). If there will be many users accessing the spooled printer, the person's initials may be added for further identification of the output. A routine is written to see if a spool file of that name exists, and increments the number if it does:

Example:

```
100   DIM Spool$[160],Id$[3]
110 !
120   ! Spool$ is of the form <directory path>[:<msus>]
130 !
140   Spool2631:REMOTE" !Name of Spool Directory on this SRM
150   LINPUT "What are your initials (optional for printout id)?",ID$
      .
      .
      .
1000  Pos=POS(Spool$&":",":") ! If there is no msus in Spool$, Pos
1010 !                          points past the end of the string
1020 !
1030 ! Spool$[1,Pos-1]   contains the spool directory pathname
1040 !
1050 ! Spool$[Pos]       contains the mass storage unit specifier (or
1060 !                   a null string if a msus is not in Spool$
1070 !
1080  Count=1
1090  ON ERROR GOTO No_file
1100 Loop: ASSIGN #5 TO Spool$[1,Pos-1]&"/TEXT_"&Id$&VAL$(Count)&Spool$[Pos]
1110  Count=Count+1      !If got here, file already exists - try again
1120  GOTO Loop
1130 No_file: CREATE ASCII Spool$[1,Pos-1]&"/TEXT_"&Id$&VAL$(Count)&Spool$[Pos],10
1140  ASSIGN #TO Spool$[1,Pos-1]&"/TEXT_"&Id$&VAL$(Count)&Spool$[Pos]
      .
      .     (Print routine goes here)
      .
2000  ASSIGN #5 TO *  !Close spool file so it can be printed
```

Printing to a Spool File

The easiest way to implement printer spooling in a program is to set up a flag that indicates whether a spool file or a printer is being used, and to write a short subprogram to choose between methods. The following example uses the flag (SRM) which is 0 when using a printer and 1 when using a spool file. File #5 is assumed to be the spool file as in the previous example.

Example:

```
1500  CALL Print(#5,Srm,"Final Results:")
1510  FOR Place=1 TO 10
1520     CALL Print(#5,Srm,Name$(Place))
1530  NEXT Place
      .
      .
      .
5000  SUB Print(#1,Flag,Print$)
5010  IF NOT Flag THEN PRINT Print$
5020  IF Flag THEN PRINT #1;Print$
5030  SUBENT
```

Formatted Output With an I/O ROM

Formatted output can be accomplished by setting up a string variable and outputting to the string, then printing the string to the printer or the spool file.

Example:

```
10    DIM Output$[80]
      .
      .
      .
1500  OUTPUT Output$ USING "#,34X,K";"Final Results"    !Center title
1510  CALL Print(#5,Srm,Output$)
1520  FOR Place=1 TO 10
1530     OUTPUT Output$ USING 1540;Place,Name$(Place)
1530     OUTPUT Output$ USING 1540;Place,Name$(Place)
1540     IMAGE #,DD,5X,40A
1550     CALL Print(#5,Srm,Output$)
1570  NEXT Place
      .
      .
      .
5000  SUB Print (#1,Flag,Print$)
5010  IF NOT Flag THEN PRINT #1;Print$
5030  SUBENT
```

Note

If the carriage return/line feed suppression # is not used, a carriage return and linefeed will be printed as the last two characters in the string variable, causing double-spaced output.

Formatted Output Without an I/O ROM

The only way to accomplish formatted output without the I/O ROM is to build a string.

Example:

```
1500  IF Srm THEN Build
1510  PRINT TAB(35);"Final Results"
1520  FOR Place=1 TO 10
1530     PRINT Using 1540;Place,Name$(Place)
1540     IMAGE #,DD,5X,40A
1550  NEXT Place
1560 GOTO Continue
1570 Build: PRINT #5;RPT$(" ",34)&"Final Results"
1580  FOR Place=1 TO 10
1590     Output$=RPT$(" ",2-LEN(VAL$(Place)))!Leading space if Place<10
1600     Output$=Output$&VAL$(Place)&"      "&Name$(Place)
1610     PRINT #5;Output$
1620  NEXT Place
1630 Continue: !
```

Chapter 5
HP 9845 Workstation Utilities

Along with your Shared Resource Management System, you received a tape called "Resource Management Utilities", which contains these programs:

- MENU
- 98029A
- PURGE
- VOLCAT
- BACKUP
- RESTOR

and these binaries:

- BDAT
- PLOTSP

Each utility can be run by LOADing it and then pressing the [RUN] key. Alternately, you can LOAD the MENU program which lists all of the other utilities with a number next to each. Each program can then be run merely by entering the appropriate number. If you turn on your HP 9845 with the utility tape inserted and the [AUTO ST] key pressed in, the system powers-up by loading and running the MENU program.

The MENU Utility

This Utility lists the other Utilities with a number next to each one. After you select one of them, the system asks you whether you want to run the Utility or whether you want a description displayed. If you press 1, the system displays a brief description of the specified Utility. If you press 2, the system LOADs and RUNs the specified Utility.

The 98029A Utility

This Utility performs a diagnostic test of all HP 98029A interface cards connected to your Shared Resource Management System. It then performs a roll call of all other interfaces, listing the device type and node address. Finally, the HP 98029A Utility performs an exercise on all connected system controllers and displays a message stating whether they are functional or not.

The PURGE Utility

This Utility enables you to purge all files and directories in a particular branch of a directory tree. The system will ask you for the directory specifier of the directory at the top of the branch and the Shared Resource Management specifier. The system will also ask you for a branch password. This password is used for any files or directories on the branch that are PURGE-protected. If the password is wrong, the file or directory will not be purged. Also, files that are open or locked will not be purged.

You can choose either "verify" or "auto" mode. In verify mode, the system will ask you to verify each purge. In auto mode, the system will execute all purges without asking for verification.

If any files or directories cannot be purged, then their superior directory cannot be purged either. If the system successfully purges everything on a branch, it will also purge the directory at the top of the branch.

The VOLCAT Utility

This Utility will prompt you for a Shared Resource Management address and then list all disc volumes at that address. The unit number, volume name, and number of bytes are displayed for each volume.

The BACKUP Utility

Use this Utility to copy a directory branch, or branch segment, from a Shared Resource Management disc to a local media. The Utility allows you to specify a date to be used as a copy key. This means you can direct the system to copy only those files that were last updated before or after a certain date. You can also specify which file types (PROG, ASCII, DATA, and DIR) should be copied, and how many levels deep the Utility should go. All copied files and directories must fit on a single tape and the total number of files and directories must be less than 100. Passwords are NOT copied.

The RESTOR Utility

Use this Utility to copy files and directories from a local tape to a Shared Resource Management disc. The files and directories must have previously been copied to the tape with the HP 9845 BACKUP Utility. As with the BACKUP Utility, you can specify a key date, file types, and level depth. Passwords are not restored.

HP 9845 Workstation Binaries

The BDAT and PLOTSP binaries must be loaded individually. These binaries extend the basic capabilities of your HP 9845.

The BDAT File Access Binary

The BDAT file access binary enables your HP 9845 workstation to access BDAT files created by the HP 9000 family: Series 200 (Models 16, 26 and 36) and Series 500. Since these files are generally compact and also permit both sequential and random access, your HP 9845 can share data more efficiently with HP 9000 workstations.

The term "BDAT" refers to files created as BDAT type files by the HP 9000 family. BDAT files store data in IEEE binary formats.

Note

Note that these BDAT files are NOT the same as the HP 9845 created BDAT files, which are created and accessed with the FCREATE, FREAD or FPRINT statements.

All access to files referred to in this document is made through the Shared Resource Manager (SRM). This document uses as a foundation the concepts found in the HP 9845 Operating and Programming manual, part number 09845-93000, and the SRM Operating System Manual, part number 98619-90030.

This binary requires only that the SRM ROM for the HP 9845 be present; it works independently of all other option ROMs and binaries and with any read/write memory configuration. You also must have a connection to an SRM.

Converting a PROG File

If you created a PROG type file with your HP 9845, you must first re-syntax the file to allow the BDAT binary to use certain mass storage keywords. Note that this conversion is necessary only for PROG type files. Follow these steps:

1. Load the PROG type file into the internal memory by typing:

   ```
   LOAD "Prog_file"
   ```

2. Save the Prog_file as a DATA or ASCII file by typing:

   ```
   SAVE "Data_file"
   ```
   (saves an DATA file)

   or

   ```
   SAVE ASCII "Ascii_file"
   ```
   (saves an ASCII file)

3. Get the Prog_file again by typing:

   ```
   GET "Data_file"
   ```

   or

   ```
   GET "Ascii_file"
   ```

4. Store converted file on your mass storage media by typing:

```
STORE "Converted_file"
```

or

```
PURGE "Prog_file"
STORE "Prog_file"
```

The conversion of your PROG file is now complete.

Loading and Executing the BDAT Binary

To load and execute this binary type:

```
LOAD BIN "BDAT:T15"        EXECUTE
```

The BDAT binary checks to determine whether the binary has already been loaded; if so the most recent version is deleted.

Using the BDAT Binary

The following examples illustrate using the BDAT binary to access BDAT file types through the SRM. Optional parameters are indicated with square brackets: [optional_param], N can be an integer in the range of 1 to 10 and R indicates a defined record number for random access.

You can open and close BDAT files on your HP 9845 with:

```
ASSIGN #N TO "Bdat_file" [,Return]
ASSIGN "Bdat_file" TO #N [,Return]
ASSIGN * TO #N
ASSIGN #N to *
```

To write data to a BDAT file use:

```
PRINT #N [,R]; Data_list
MAT PRINT #N [,R]; Array_variable_list
```

To read data from a BDAT file use:

```
READ #N [,R]; Variable_list
MAT READ #N [,R]; Array_var_list [(Redim_subscripts)]
```

All syntax is identical to the syntax used for DATA and ASCII files; no program syntax changes are necessary.

Record Lengths

The HP 9000 Series 200 and 500 workstations allow logical record lengths greater than 32 767 bytes. Since you cannot use these larger length records on your HP 9845, an error (number 133) results if you attempt to open these files.

A record length of one byte is allowed permitting byte addressing. Random accesses to byte stream files position the pointer to the byte specified, and then it is treated in a serial manner. Note that byte stream files can have a maximum length of 32 767 bytes.

For all other files with record lengths greater than one byte, the record length must be even (evenly divisible by 2). Random accesses to these files that attempt to cross a record boundary result in error 60. Serial accesses may cross record boundaries. Strings of an odd length are padded to an even length.

The HP 9000 Series 200 and 500 workstations also allow more than 32 767 records per file. The HP 9845 supports a maximum of 32 767 records per file; therefore error 59 results if the HP 9845 tries to serially access data in records past this limit.

BDAT Formats

BDAT files are streams of binary numbers and do not store information about the data type. No coercion of numerics from one type to another is therefore possible. When you read a BDAT file you must know the format of the data to interpret the data correctly.

The BDAT binary automatically converts between the BDAT IEEE binary format and the BCD format used by the HP 9845.

Since only the HP 9000 Series 500 supports a DOUBLE integer format, and the HP 9000 Series 200 does not support a SHORT data type, the BDAT binary supports only INTEGER, REAL and STRING data types. Error 132 results if you attempt to use a SHORT data type in a BDAT file on the HP 9845.

Precision

INTEGERs are identical in format on the HP 9845 and HP 9000 Series 200 and Series 500. They are fully interchangeable with no loss of precision.

REALs are represented differently on the HP 9845 and the HP 9000s. REALs printed by the HP 9845 have 12 digits for the mantissa; the exponent is restricted to the range of −99 through +99. If you attempt to read or print data with exponents outside this range, a precision overflow error occurs (error number 22).

STRINGs must be restricted to a length of 32 767 to be interchangeable. An attempt to read a string longer than this limit results in an error (error number 18).

Some accuracy may be lost as a result of conversions between binary and BCD formats. This is a result of rounding errors and precision differences. Conversion to or from binary format is accurate to within a one bit rounding error.

Performance

Printing REALs is four to five times slower to BDAT files than to DATA files; while reading REALs is nine to ten times slower. This is due to the extra time needed to convert formats. Real numerics in the range 1E-64 to 1E+64 translate faster than those outside this range.

The BDAT file uses half as much space per INTEGER as does a DATA file. Therefore, even though the bytes per second transferred is lower with BDAT files, the actual INTEGER data per second transfer rate is higher.

BDAT files use a four byte length field for STRINGs while DATA and ASCII files use two bytes. In addition, DATA files have other header fields. The transfer rate for STRINGs is comparable to that of other files.

Errors

The BDAT file access binary for the HP 9845 adds two new error numbers to the BASIC error numbers:

```
132     SHORT numeric type not supported for BDAT files.

133     Record size > 32767 bytes.
```

The message "RESOURCE MANAGEMENT ROM MISSING" is displayed if the binary is present without the resource management ROM.

The BDAT binary also uses the following existing BASIC errors:

```
22      Real precision overflow.

51      File not currently assigned.

58      Improper file type. You can assign files of type DATA,
        ASCII and BDAT.

59      Physical or logical end-of-file found.

60      Logical end-of-record found in random mode.
```

The PLOTSP Binary

This Utility permits the HP 9845 desktop computer to send graphics output to files that can be spooled to shared plotters. All access to files referred to below is made through the SRM Controller.

Loading the PLOTSP Binary

The Plotter Spooling Binary, PLOTSP, is an optional BINARY program for the HP 9845. This binary requires both the SRM ROM and the graphics ROM to be present. It works independently of all other option ROMs and binaries and works with any R/W memory configuration. A connection to a SRM Controller is required. The binary is approximately 2.5K bytes in size, and removes that many bytes from available user R/W memory. No additional memory is needed for system requirements. The binary is loaded by typing:

```
LOAD BIN "PLOTSP:T15"  EXECUTE
```

Using the PLOTSP Binary

After the PLOTSP binary is loaded, you can send plots formatted in Hewlett-Packard Graphics Language (HPGL) to a file for spooling to an HPGL-compatible shared plotter. The binary adds two new Basic-language statements to the HP 9845 workstation to support plotter spooling:

- PLOTTER IS #
- CLOSE PLOTTER #

The PLOTTER IS # statement opens the file specified, associates the file number with the open file, and prepares the file contents for receiving plotter commands.

The CLOSE PLOTTER # statement closes the file. This initiates the spooling activity if that file is located in a spooling directory on the SRM.

An example of the use of the PLOTTER IS # statement with a numeric variable and a string variable is:

```
PLOTTER IS #File_number,File_specifier$
```

You may also include the resolution if the default value of 0.025 cm is to be changed, such as:

```
PLOTTER IS #4,File$&Msus$,Resolution
```

Notice the concatenation of two file specifiers. You may decide to include the number of pens, such as 6 (default = 8). If you do, the resolution **MUST** also be included in the statement, as in the following:

```
PLOTTER IS #3,"FRED",0.025,6
```

An example of a program segment is:

```
10 DIM F$[80]
20 F$="PLOTSPOOLDIR/MYFILE:REMOTE"
30 CREATE ASCII F$
40 PLOTTER IS #1,F$
50 !  PLOT STATEMENTS GO HERE
60 CLOSE PLOTTER #1
70 END
```

The SRM spooler process expects the plot file to be an ASCII file. Using a DATA file, or any file other than an ASCII file, causes the spooler to misinterpret the file contents and produce indeterminate results.

The ASCII file which receives your graphics output must be CREATEd (see statement on line 30 above) before executing the PLOTTER IS # statement. The ASCII file may be created in the spooling directory, in which case spooling begins as soon as the file has any data and is closed. The spooling process plots your data and then purges your file. If you wish to retain a copy of your file, create it in some other directory and then copy it to the spooling directory using COPY.

Do not use an ASSIGN # statement to re-assign the file_number, or subsequent plots will be sent to the new file.

```
10 PLOTTER IS #1,"PLOT1"
20 ASSIGN #1 TO "FILE_Z"
30 PLOT 50,50
```

The above example sends the PLOT 50,50 to the file "FILE_Z". Use `CLOSE PLOTTER #` or `PLOTTER IS OFF` inserted between statement lines 10 and 20 above, for example.

The graphical output is simply a series of HPGL strings. By using the `PRINT #` and `READ #` statements, you can examine and manipulate the data before plotting.

Normally, the platen area coordinates (P1 and P2) are available to the workstation from the plotter. On the SRM system, however, the plotter interaction is output only; P1 and P2 are not read by the spooler. Upon execution of the `PLOTTER IS #` statement, P1 and P2 are set for HP metric paper (38cm x 25cm). You can reset P1 and P2 by invoking the LIMIT statement, but the LIMIT parameters are accepted without checking their bounds. This may result in P1 and P2 being set outside the platen area turning on the out-of-limit light.

Devices utilizing input-only functions may not be used with a spooling plotter. The DIGITIZE and CURSOR statements, which require data from a graphics input device, generate error 38 (I/O function not allowed).

Plotting statements can only be sent to a spool file from within a program and not from the keyboard. Many of the plotting statements generate multiple lines of output which is equivalent to executing an illegal multi-line function from the keyboard.

Appendix A
Glossary

access capability — Different capabilities are required to perform different remote mass storage operations. Shared resource file protection provides three access capabilities: MANAGER, READ and WRITE. The capabilities required for each command are shown in the language reference sections of this manual. When files are first created, all capabilities are available to the public. PROTECT, a BASIC command, allows you to remove specified capabilities from public access and protect them with one or more passwords. Each file can have any number of passwords, each password allowing specific access capabilities.

ASCII file — A sequential access file of variable length records. Each record is composed of a length field (16 characers long) followed by the number of ASCII characters specified by the length field. A length of −1 signifies the end of the file. These files are compatible between the HP 9845B/C and the HP Series 200 workstations.

BDAT file — There are two different types of BDAT files. The HP 9826/9836 computers have a unique BDAT file structure that can be exchanged only with other HP 9826/9836 workstations, and HP 9845 with BDAT binary. The HP 9845 also has a BDAT file structure which is unique to that model computer. HP 9845 BDAT files cannot reside on shared discs. The FCREATE, FREAD, and FPRINT commands apply only to local HP 9845 BDAT files.

BIN file — There are two incompatible BINary file types. The HP 9845B/C computers support one type of binary file in local operation. HP 9826 and HP 9836 computers support the second binary file type both in local mode and when using the shared resource system. Different HP 9826/9836 workstations can share BIN files through the shared resource system.

directory — Directories are files that contain information about the type, location, size, etc., of the subordinate files and directories in the next level of the shared resource hierarchical directory structure. Directories cannot be copied. As with any file, directory files can be protected by passwords. Directory files are created using the BASIC language CREATE DIR command.

directory name

A directory name is a string expression from 1 through 16 characters. Legal characters include all upper and lowercase ASCII letters, numerals, the underbar (_), the period, and CHR$(161) through CHR$(254). Directory names can be made of any combination of upper and lowercase letters. Since the name is enclosed in quotes, even keywords can be used as directory names. There are two special cases. If the directory name consists of two periods (..), the superior directory to the current working directory is specified. Since the root directory has no superior directory, the double period specifies the root directory when the root directory is the current working directory. If the directory name consists of a single period (.), the current working directory is specified.

directory specifier

A list of file names which uniquely specify a directory, given a starting point in the directory structure (current MASS STORAGE IS directory) and a path to the desired directory. From BASIC this is accomplished by naming the directories or path, separating the names with a stroke (/) character. Directory specifier may not contain multi-line function calls

extendable file

All shared-resource files are extendable. When you write past the specified file extent size, the system automatically extends or enlarges the file by a size equal to the extent size specified when the file was created. Since the shared resource files are not constrained to be contiguous on the mass storage media this extensibility is possible.

file

A file is a collection of records, defined by the programmer, and accessed by the workstations as a single unit. Each file has a unique file specifier. The shared resource system supports the following:

HP 9826/HP 9836 BASIC files
- ASCII
- BDAT
- BINary
- DIRectory
- PROGram
- SYSTM

HP 9000 files
- ASCII
- BDAT
- BINary
- DATA
- DIRectory
- PROGram

HP 9845B/C BASIC files
- ASCII
- DATA
- DIRectory
- PROGram

HP 9826/9836 Pascal Files
- TEXT
- CODE
- DATA

file name

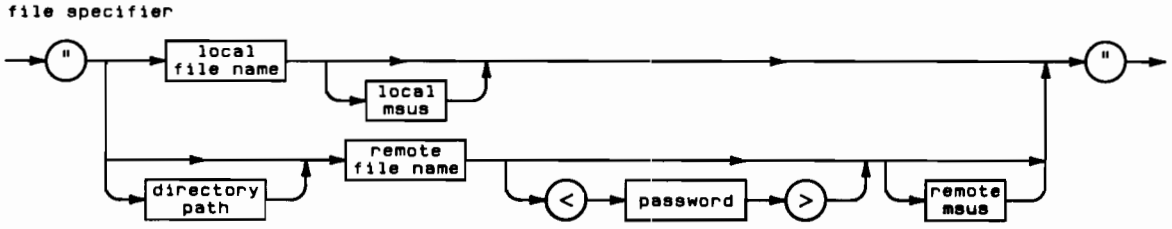
The name of a file. The table below indicates the size and allowable characters for file names used with the different workstations and modes of operation of the shared resource system. To copy files between different workstations and between local and remote on a single workstation may require file name changes.

Mainframe/ Mass Storage	Name Size (Characters)	Usable Characters	ASCII Code (decimal)
HP 9826/HP 9836 and HP 9845 files stored on the shared resource system.	1 to 16	Uppercase Letters	65 thru 90
		Lowercase Letters	97 thru 122
		Digits (0 thru 9)	48 thru 57
		Graphic Characters	223 thru 254
		Period	46
		Underscore	95
		Foreign/graphic characters	161 thru 254
HP 9826/HP 9836 files stored on local mass storage	1 to 10	Uppercase Letters	65 thru 90
		Lowercase Letters	97 thru 122
		Foreign Characters	161 thru 222
		Digits (0 thru 9)	48 thru 57
		DO NOT USE:	
		Control Characters	0 thru 32
		Pound (#)	35
		Asterisk (*)	42
		Comma (,)	44
		Colon (:)	58
		Equals (=)	61
		Question Mark (?)	63
		Left Bracket ([)	91
		Right Bracket (])	93
		Del	127
HP 9835/HP 9845 files stored in local mass storage	1 to 6	Any Character	
		DO NOT USE:	
	Colon	58	
		Quote Mark	34
		Null	0
		Del	255

file specifier

A string expression that identifies a unique file. File specifiers for BASIC are made up of a file

name and msus in local mode and a directory path, file name and msus for shared resource (remote) operation. File specifiers depend on both the mass storage device and workstation. The syntax diagram for the file specifier is shown below:

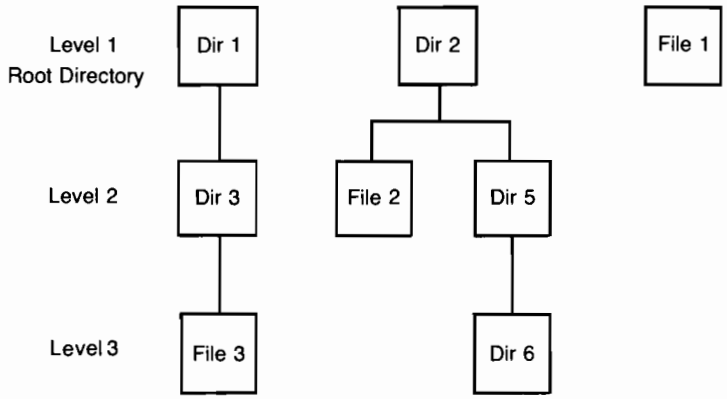


interface select codes

See select codes.

level

The hierarchical directory structure of the shared resource management system can be viewed as a layered structure. The Root directory is at level zero and each following directory structure follows on level one, two, etc. Each directory, except the root directory, has a superior directory located one level closer to the root directory and the possibility of at least one subordinate directory located one level further away from the root directory.



line numbers

The HP 9845B/C allows BASIC line numbers from 1 thru 32 766. The HP 9835A/B allows BASIC line numbers from 1 thru 9999.

local

The term local is used to describe workstation operation in a stand-alone configuration. This is opposed to remote (shared resource) operation where shared peripherals and shared disc drives are used.

locked file

The BASIC keyword LOCK provides the user with sole access to a file. Trying to lock a locked file from any workstation other than the station that originally locked the file returns a message stating that the file is not accessible. When making a READ or WRITE request of a file locked by another workstation, your workstation will wait for the file to be unlocked before completing the operation.

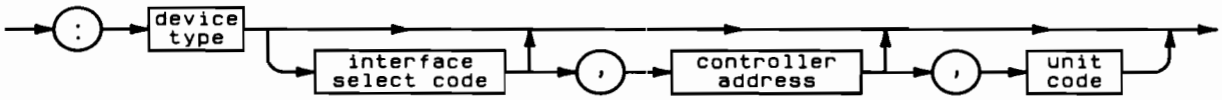
MANAGER access

This access capability allows the user to purge, rename, assign passwords, and read and write files and directories.

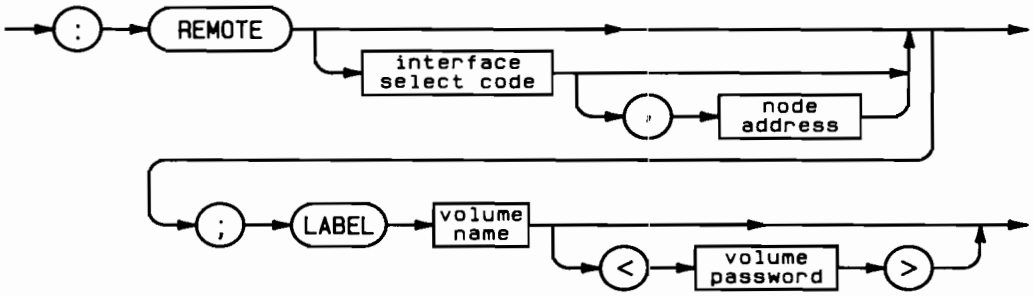
msus

This is the acronym for "Mass Storage Unit Specifier". It is a string expression that specifies a device to be used for a mass storage operations. Syntax drawings for both local and remote specifiers are shown below. Any path from the left entry point into the drawings to the exit point on the right of the drawings (without going against the arrows) gives a correctly syntaxed msus.

local mass storage unit specifier for HP 9845B/C



remote mass storage unit specifier



node A node is a point of origin or destination for any information transfers between workstations in the system. Information transfers are always between workstations and the shared resource management controller, never between workstations. Node addresses are assigned to each shared resource management system interface in the system.

node address Each HP 98029 and HP 98629 shared resource management system interface has a node address assigned to it. Each information packet transmited between system computers contains both the source and destination node address for the information contained in that packet.

opened file To read or write to a file, it must be opened. A file is opened from BASIC with the ASSIGN statement.

packet A packet is a set of information transmitted as a single unit between workstations. It contains control information, source and destination node addresses, and data or other information pertaining to the system operation.

password A string of ASCII characters used to limit access capabilities to shared resource files and directories. A password is required by the shared resource security system before it will provide non-public capabilities or files. Passwords can be from 1 to 16 characters long.

peripherals See the "SRM Supported Peripherals" Appendix for a list of peripherals that are supported by the shared resource management system.

protected capability An access capability (READ, WRITE or MANAGER) is said to be protected if it has been assigned one or more security passwords. The ability to use the specified capability is then limited to those users who specify the password. Capabilities that are not protected are called public capabilities.

public capability A capability (WRITE, READ or MANAGER) is said to be public if it has not been assigned any passwords. That is, any user has access to public capabilities.

READ access This access capability allows the user to read a protected file, catalog a protected directory, or pass through a protected directory to a lower level. This capability is also required in addition to WRITE access to create or purge a file or directory within another directory.

record Records referred to in this manual are defined records. They are the smallest unit of information accessible on a mass storage media. The length of a record is determined by the CREATE statement or is fixed by the system.

remote The term remote is used to describe operations that use shared peripherals and disc drives rather than local peripherals and resources that are connected to the individual workstations.

resource management multiplexer (HP 98028A) An electronic switching device that controls data transmission between workstations and controllers in the shared resource management system.

root directory Each disc volume has a root directory which is the highest directory in the shared resource hierarchical directory structure. That is, all directories are subordinate to the root directory. The root directory is the only directory on level 0. All other directories in the structure are at lower levels; i.e., level one, two, etc. The root directory has no name (all blanks), but its name can be treated as being the same as the volume name.

select codes

9845 = 0 reserved for internal printer
1 thru 12
13 reserved for internal graphics
14 & 15 reserved for tape drives
16 reserved for CRT

98029 = 1 thru 12 for the HP 9845; 1 thru 14 for the HP 9835

9835 = 0 reserved for internal printer
1 thru 14
15 reserved for tape drive
16 reserved for CRT

shared file A file which can be concurrently accessed by multiple workstations.

shared resource management system A multi-user system through which multiple workstations (HP 9826/9836 and/or HP 9845B/C) can share disc drives and peripherals.

shared peripherals Disc(s) and printer(s) connected to the shared resource management system controller that are shared among the workstations in the system.

spooling A process by which an ASCII file can be written into a special "spooling" directory, which is associated with a particular printer. When the printer is available, the file is printed and then purged from the directory. Once the file is written to the spooler directory, the workstation is free to perform other tasks.

resource management controller — The HP 9826 that controls shared disc and printer resources, and controls access to them by the workstations in the Shared Resource Management System.

SRM — Acronym for "Shared Resource Management".

shared resource management operating system — The software that enables an HP 9826 or HP 9920 computer to act as a Shared Resource Management System Controller.

subordinate directory — A directory whose name appears in a catalog listing of another (superior) directory is said to be subordinate to that superior directory.

superior directory — The directory which lists a file or directory in its catalog listing is the superior directory to that file or directory. The root directory has no superior directory.

system map — A diagram showing a detailed pictorial representation of the standard shared resource system, including node addresses, equipment locations, equipment types, etc. A system map for your system should be located near your system's controller.

system unit numbers — User-selected identifiers that the shared resource management system controller uses to uniquely identify shared peripherals and interfaces.

time stamp — Each file in a directory is given a time stamp indicating the year, month, day, hour and second of the last update to that file.

unit numbers — See system unit numbers.

volume — The term volume refers to a mass storage media. Many disc drives contain one volume of storage which may contain many files. Each volume has a name assigned to it.

volume name — A string expression that identifies a unique volume. This expression can be from 1 through 16 characters long. Upper and lowercase alpha characters, the digits (0 thru 9), the period (.) and the underscore (_) character can be used to specify a volume name.

volume password — When a disc media is initialized, a password is assigned to protect that volume. This is a "super password" which supercedes the individual protection of any file or directory on the disc media. That is, the volume password allows access to ANY file or directory on the disc media. This password contains from 1 thru 16 upper and/or lower case alpha characters, the digits (0 thru 9) or the underscore (_) character.

working directory The directory specified by the last BASIC language MASS STORAGE IS (MSI) statement executed. If no directory was specified, the working directory defaults to the root directory.

workstation The computers used in the Shared Resource Management system. These include the HP 9826, HP 9836, HP 9845B and HP 9845C.

WRITE access This access capability allows the user to write to a file or create and purge subordinate files in a directory file.

Appendix B

SRM Error Codes HP 9835A/B and 9845B/C

The following error codes are supported by the Shared Resource Management System (SRM). Each code has a partial list of possible causes listed with the error text.

57 Mass storage ROM is missing
OR
Illegal or unimplemented resource management command
OR
Command syntaxed without the Resource Management ROM present.

HP 9845 mass storage ROM, part number 98413 is not in the computer and a mass storage statement was attempted.

HP 9835 Mass Storage ROM, part number 98331 A/B, is not in the computer and a mass storage statement was attempted.

Attempted to use unsupported mass storage command while using the SRM system.

A PROG file written on an HP 9845 WITHOUT the SRM ROMs installed cannot be used on SRM without being converted with a GET and SAVE of one file. The program must be recompiled.

120 Shared Resource Management system error.

This error seldom occurs. When it does, an error message can be found on the SRM controller's display.

121 SRM restarted or down.

If the SYSTEM DOWN statement and SYSTEM UP statements are used, OR there is a loss of power to the SRM controller, the operating system automatically closes all files. To recover, just respecify your working directory with an MSI ":REMOTE" statement.

122 Volume not found or volume I/O error.

You misspelled the volume specifier or something is wrong with the shared disc drive.

123 File or directory in use or directory not empty.

You tried to purge or rename an open file or tried to purge a directory that still had files or other directories listed in it. Also, you cannot initialize a volume with any open files stored on it.

124 File is locked or deadlock condition detected.

You cannot READ, PRINT #, LOCK, etc., files that another user has locked.

125 Remote node doesn't respond.

126 Wrong select code or I/O card failure.

127 Link is disconnected or down.

Multiplexer is not powered or something is wrong with the cables.

128 Illegal on local mass storage.

Some mass storage commands are applicable only to SRM. For example, CREATE DIR "A:T15" is not legal.

129 Not allowed for directories.

Trying to move a directory with the RENAME command is illegal.

130 Improper password or password not found.

Syntax for local and remote passwords is different. Don't mix them up.

131 Msus not allowed in destination specifier.

132 SHORT numeric type not supported for BDAT files.

133 Record size >32 767 bytes.

```
RESOURCE MANAGEMENT TIMEOUT ON SELECT CODE n
```

This is a warning, but may indicate an error. The controller is busy processing a long request or there are too many outstanding requests. Everything will proceed normally if you wait. In extreme cases, you may have to reset your workstation.

Appendix C
SRM 2.0 Supported Peripherals

The following is a list of the peripherals supported by the SRM operating system:

CS/80 Disc Drives

7908 - media backup on built-in tape
7911 - media backup on built-in tape
7912 - media backup on built-in tape
7914 - media backup on built-in tape

Line Printers

CIPER protocol printers (the HP 2608S and HP 2563A) provide excellent error handling and extremely low controller CPU overhead. These printers provide the most satisfactory system performance.

2631A - except for perforation-skip mode
2631B/G -
2608A - driver supports translation of some escape sequences for special functions
2608S -
2563A -
9876A - as a low performance printer, not recommended

Plotters

For most plotters, the "error" and/or the "out of limit" light(s) may come on during plotter initialization and after the plotter is done. This is normal. However, if either light comes on **during** the plot, the user's data is in error.

9872A - plotter does not identify itself to the controller
9872B/C -
9872S/T - paper advance works
7580A/B -
7585A/B -

Subject Index

o

p

r

s

u

v

w