4700 Personal Computer Application Services

4700 Personal Computer

Personal Computer Software

SC31-3016-1

4700 Personal Computer Application Services

4700 Personal Computer

Personal Computer Software

SC31-3016-1

Second Edition (September 1986)

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below; requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comment is provided at the back of this publication. If the form has been removed, address comments to IBM Corporation, Information Development, Department 78C, 1001 W.T. Harris Boulevard West, Charlotte, NC 28257, USA. IBM may use or distribute any information you supply in any way it believes appropriate without incurring any obligation to you.

About This Book

The 4700 Personal Computer Application Services provide communications between a personal computer and a 4700 controller or other host system. The *4700 Personal Computer Application Services User's Guide* contains information the programmer needs to write programs that use application services. It also explains other capabilities that Application Services programs make available to the user.

Section 1, "Introduction" explains some of the capabilities of Application Services.

Section 2, "Customizing Application Service Programs" explains how to customize the 4704 terminal emulation that is part of the Application Services control program.

Section 3, "Installing the Application Services at the Personal Computer" details how to install the Application Services and how to prepare the operating disk or diskette.

Section 4, "Transferring Files" explains how to transfer complete files between the personal computer and the 4700 controller or host.

Section 5, "Using Function Calls" details function calls that application programs running in the personal computer can use to communicate with the host controller.

Section 6, "Using Virtual Volumes" explains how to use the virtual volume (VVOL) facility. The VVOL facility allows the personal computer to access 4700 controller disk files as if they were DOS disk/diskette files.

Section 7, "Diagnosis Guide" explains how to analyze and report problems with the personal computer.

Section 8, "Using the High Level Language Interface" contains details on the high level interface facility. This facility allows application programs in high level languages to access DOS and Application Services functions. A detailed table of contents begins this section.

Appendix A, "Controller File Transfer Program" explains the installation and use of the file transfer program for the 4700 controller. This program is the counterpart of the file transfer programs described in Section 4, "Transferring Files."

The glossary defines new and previously defined terms found in this book.

Use the index to cross-reference terms and subjects.

Other Books to Read

Refer to the following publications for more information on subjects covered in this book:

4700 Finance Communications System, Controller Programming Library,

- *Volume 2: Disk and Diskette Programming*
- *Volume 4: Loop and DCA Device Programming*

4700 Finance Communications System, System Summary

IBM 3270 Data Stream Programmer's Reference

Disk Operating System reference for your personal computer.

DOS Technical Reference for your personal computer.

Macro Assembler reference for your personal computer.

4700 Personal Computer Financial Input Adapter Device Driver Microcode User's Guide

4700 Personal Computer Financial Output Adapter Device Driver Microcode User's Guide

IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System.

Contents

Section 1. Introduction

A personal computer communicates with the 4700 system using the functions and commands of the Application Services and the 3278/79 Emulation Adapter card. The 3278/79 Emulation Adapter card provides a DCA (device cluster adapter) data link, which attaches to a DCA port on the 4700 controller.

The Application Services programs and the 3278/79 adapter provide the following capabilities:

- Personal computer emulation of a 3278/4704 terminal
- Public and private virtual volume access
- Disk and diskette file transfer to and from the 4700 controller or to and from a host system
- Communication between personal computer application programs and the 4700 controller
- A high level language interface to DOS and Application Services function calls.

Terminal Emulation

To load the terminal emulation program enter the following line at the DOS prompt:

[d:][path]CCC

d: and **path** are the optional identifiers for the DOS drive and directory where the CCC.COM program resides.

The personal computer emulates a 3278 Model 2 or a 4704 Model 2 (1920-character screen) in controller tracking mode when the terminal emulation program is active and you press the Alt-Esc key combination. When you press Alt-Esc, the emulation program suspends the DOS application (saving both screen and storage) that is running and switches the personal computer to terminal emulation. When you press Alt-Esc again, the terminal emulation program restores the DOS application exactly where you stopped it. This process is called **mode switching.**

Mode switching is inhibited (the Alt-Esc combination is ignored) if the initialization of CCC.COM is incomplete or when 4700 input or output is pending.

Note: If the DOS application program does not use calls to BIOS (Basic I/O System, a part of DOS) to set up the display in the graphics mode, the terminal emulation program cannot restore the correct screen contents after mode switching.

The following key functions occur only if data stream mapping (DATSM) is active in the 4700 controller. See "Chapter 4: Processing 3270 Data Streams" of the *4700 Finance Communications System, Controller Programming Library, Volume 4: Loop and DCA Device Programming* manual for details on DATSM.

Ins: Insert mode is like the insert mode of the 3278. When entering a character, you insert that character instead of overstriking it. Pressing RESET (SCROLLOCK on the personal computer keyboard) returns the emulation to overstriking.

Del: The delete key deletes the character at the cursor.

See the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System* for more information on terminal emulator key customization.

Remote Initial Program Load

A personal computer that has the financial input adapter and 3278/79 Emulation Adapter cards installed can use remote initial program load. If the personal computer does not have a bootable disk or diskette, a remote IPL is automatic. If the personal computer has a bootable disk, you can opt to use remote IPL instead of normal IPL. The personal computer loads DOS and other files (or programs) from the controller if the controller has data sets (controller disk files) designated for remote IPL to use. See "Section 3. Using Remote IPL" in the *4700 Personal Computer Financial Input Adapter Microcode User's Guide* for more information on remote IPL.

Virtual Volumes

A personal computer running the Application Services control program can read and write files from data sets in the 4700 controller as if they were DOS disk/diskette files. A personal computer, specially designated as the public volume manager, can create and modify 4700 data sets so that other personal computers can use them.

File Transfer

A personal computer running the Application Services control program can download files from and upload files to the 4700 controller and the host system. The SEND and RECEIVE commands provide this function.

Application Services Function Calls

Application Services provide function calls that enable application programs running in the personal computer to communicate with the controller. These function calls are in the DOS format. They allow an application program to simulate the keystrokes and functions of a terminal connected to the host controller.

Translation Function Calls

The ASCII/EBCDIC translation function calls allow you to convert data from ASCII to EBCDIC and from EBCDIC to ASCII according to one of a customizable set of translation tables.

High Level Language Interface

The HLLI (high level language interface) programs enable applications in one of several high level languages to use DOS, Application Services, and translation function calls. Also, you can use the HLLI profile to define high level language interfaces to new interrupt handlers.

Running Personal Services_PC

You can use Personal Services/PC with 4700 Personal Computer Application Services. Personal Services/PC is an IBM program that lets your personal computer communicate with users who are part of a host office system network, such as the Distributed Office Support System (DISOSS). Personal Services/PC allows you to send, receive, and catalog electronic mail within your host office system. You must purchase Personal Services/PC separately; it is not part of the 4700 Personal Computer Application Services.

When you install Personal Services/PC in a personal computer that also has Application Services installed, identify the personal computer as a 3270 Personal Computer. During its installation process, Personal Services/PC displays an installation panel (PS/PC:INS) that asks you to select the host attachment you are using. Enter a "2" to indicate a 3270 Personal Computer attachment. You must identify the personal computer running Application Services as a 3270 Personal Computer in order for Personal Services/PC to run.

Personal Services/PC also provides a panel (PS/PC:9) where the communication equipment defaults may be reviewed or changed. Since Application Services does not support multiple host sessions, there is no need to modify the default 3270 Personal Computer connection short name given here.

Personal Services/PC is not compatible with the Application Display Management (ADMS) program product. For this reason you should not attempt to run the Application Services customization program while PS/PC is installed.

There might be other considerations if you install Personal Services/PC on a personal computer that is attached to a 4700 controller using the Passthru Support of the Resource Manager for the 4700. See these books for more information:

- *Resource Manager for the 4700 Finance Communication System: Programmer's Guide*
- Personal Services/PC *Technical Reference*
- *Distributed Office Support System/370 Version 3 Release 2 Personal Services/PC Support*
- *How to Use Personal Services/PC.*

Section 2. Customizing Application Service Programs

The application services customization command, CUSTOM.BAT, allows you to customize the 4700 Personal Computer facilities, and to configure your personal computer's operational characteristics, such as which device drivers will be active.

Before you customize the Application Services program, use the DOS DISKCOPY command to copy the Application Services diskettes. Then save the original diskettes and use the copies. See the IBM Personal Computer *Disk Operating System* manual for details about using DISKCOPY.

You may need several different operating diskettes, each with a different customized version of the application services programs or other 4700 Personal Computer programs. Copy your application services diskette for each new version before you run the customization program.

The steps in customizing application services are:

- Run the CUSTOM program.
- Run the CREATE program to create a new operational diskette.

For details on the CREATE program, see Section 3, "Installing the Application Services at the Personal Computer."

Running the Customization Program

CUSTOM.BAT is the name of the Application Services customization batch file. To run the customization program, make a diskette drive the default drive and insert the application services diskette 1. Then enter the following at the DOS prompt:

```
CUSTOM
```

Run the customization program in the default drive so that it can access other programs and files in the default drive.

The customization program displays a menu that allows you to customize any of the following options:

Option	Customization
Keyboards	Customizes any of the attachable 4700 keyboards for use in PC/DOS mode only. (See the 4700 Personal Computer Addendum to the 4700 Finance Communication System for a description of customizing your emulator keyboard.) The user defines the keyboard translation files for the KEYBD.COM command (part of the financial input adapter microcode). This program prompts for insertion of the financial input adapter diskette and creates a new keyboard translation file.
Printer	Customizes characteristics of a 4710, 4715, or 4720 printer attached to the personal computer using the Financial Output Adapter. The program prompts for insertion of the financial output adapter microcode diskette and will create

a modified copy of the printer device driver.

MSR/E Selects the MSR/E CANCEL key. This process prompts for insertion of the financial input adapter microcode diskette.

PIN Selects the PIN Pad CANCEL key. This process prompts for insertion of the financial input adapter microcode diskette.

ASCII/EBCDIC Translate Tables
Customizes up to seven ASCII to EBCDIC and EBCDIC to ASCII translation tables for use with the translation function call.

Color Display Emulation
Customizes 3278 color attributes for use when running in 3278 emulator mode. Other customization of keyboards or displays in 3278 emulation requires use of the 4700 CPGEN macros INTRTBL and OUTRTBL. See the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System* for additional information on 4700 CPGEN macros for use with application services.

Present Keystroke
Customizes the ASCII-to-4700 scan code translation table that is used by the CCC.COM program when the 6Fh Present Keystrokes (function 8) function call is used.

Other Allows the user to run any additional customization programs.

Operational Diskette

Selection of what to include on the operational diskette. The CREATE

command uses this information when building the operational diskette.

Section 3. Installing the Application Services at the Personal Computer

To install the Application Services and/or any device drivers, you create an operational disk or diskette which you can then use to start up your system. This disk/diskette must include the configuration file called CONFIG.SYS and any required files from the Application Services or device driver diskette.

To create the operational disk/diskette for the 4700 Personal Computer Application Services you will need DOS, the Application Services diskette, and the device driver diskettes for any device drivers you want to include.

Creating the Operational Disk or Diskette

To generate an operational disk/diskette, follow the steps below:

- Format a DOS system disk/diskette.
- Customize Application Services.
- Copy necessary files onto the disk/diskette.
- Use the CREATE command, and follow its instructions.

Formatting the Disk or Diskette

Use the DOS FORMAT command with the /S option to format your disk or diskette.

See the IBM PC *Disk Operating System* reference for details on the FORMAT command.

Customizing the Application Services Programs

If you have not done so already, customize the Application Services programs. See Section 2, "Customizing Application Service Programs" for details on the customization procedure. You must select the Operational Diskette option.

Copying Files to the Disk or Diskette

The Operational Diskette option of the customization procedure creates the files CREATE.BAT and OPER.BAT. Use the DOS COPY command to copy the files onto your formatted disk or diskette. Both files are necessary for the creation of the operational disk/diskette, but may be erased when the CREATE command has completed. Note that if you are using a fixed disk, your current directory should be the root directory when you copy these files.

Using the CREATE Command

Use the CREATE command to make an operational disk/diskette from your formatted disk/diskette, which should now include the CREATE.BAT and OPER.BAT files. To run CREATE, make a diskette drive the default drive and put the diskette in it; if you are using a fixed disk, make the root directory your current directory. Then enter the following line at the DOS prompt:

CREATE d1: d2:[path]

CREATE is the command that builds the operational disk/diskette.

d1: is the DOS drive specifier for the drive that contains the operational disk/diskette you are building.

d2:[path] is the DOS drive and optional directory specifier for a work drive; you will use the work drive to copy files from other required diskettes to your operational disk/diskette. If it is a diskette drive, insert diskettes as the program prompts you. If you are using the fixed disk as the work drive, be sure that all the required files are in the specified directory.

Specify both drive identifiers. If you use the same physical drive for both diskettes, you must specify two different drive identifiers. For example, if you have a single-drive system, you should specify drives A: and B:. You need to swap the operational diskette with the other required diskettes when DOS prompts you. You can use a fixed disk as the operational disk/diskette or as the work disk (or both).

After you have finished the CREATE command, your operational disk/diskette contains all the necessary files that you specified during customization. You can erase the CREATE.BAT and OPER.BAT files at this time.

Note that after you have created an operational disk, only the CONFIG.SYS file is required in the root directory. You may copy the other files to other directories as you wish, but you must then change the corresponding DEVICE= statements in CONFIG.SYS to indicate the new directory. The CONFIG.SYS file created for you by the customization procedure assumes these files will be located in the root directory at system startup.

Completing the Operational Disk or Diskette

If you want your operational disk/diskette to load the Application Services functions and communications code (as used by the SEND and RECEIVE commands) automatically, create an AUTOEXEC.BAT file and include this line:

[d:][path]CCC

This line automatically calls CCC.COM when you use the operational disk/diskette to start the system. CCC becomes resident with DOS and provides the Application Services function calls and communication functions.

See the *Disk Operating System* reference for details on the AUTOEXEC.BAT file.

If you want your operational diskette to automatically load the ASCII/EBCDIC translation functions, include this line in the AUTOEXEC.BAT file:

[d:][path]XLATE [d:][path] [,n,...]

This line automatically calls XLATE.COM, which becomes resident with DOS and provides the ASCII/EBCDIC translation functions. Replace **n** with one or more translation table numbers (1-7, created during customization) that the ASCII/EBCDIC conversions will use. Replace **n** with 0 to load the default translation table.

For both the XLATE command and the operand [,N,...], [d:] is the optional drive identifier, and [path] is the optional directory specifier for the drive and directory. If you do not specify the path and drive for the command, the XLATE.COM program must be located on the default drive and in the default directory. Use an asterisk ("*") to specify the default drive and directory for the operand.

You specify that multiple translation tables are to be loaded by including more than one N value, separated by commas. You select a specific translation table with the translation function call.

If you do not use the AUTOEXEC facility to load the CCC or XLATE code during IPL, you need to invoke the code at the DOS prompt in order to use the Application Services functions or ASCII/EBCDIC translation. The line you enter will be the same as that shown above.

Using the Operational Diskette

To use your new operational diskette, perform an IPL with the operational diskette as the IPL diskette. The IPL process examines the diskette's CONFIG.SYS file and loads the required device drivers, programs, and data into the working storage of the personal computer. After the IPL process is complete, you can remove the operational diskette.

Some Application Services functions require that you insert a diskette into a diskette drive. These are the SEND and RECEIVE commands (for file transfer) and the VVOL command (for virtual volume access). These commands are disk-resident programs that you run from the DOS command line. See Section 4, "Transferring Files" and Section 6, "Using Virtual Volumes" for more information on these commands.

Section 4. Transferring Files

Use the **SEND** and **RECEIVE** commands to transfer files between the personal computer and the 4700 controller or between the personal computer and a host system with one of the following 3270 personal computer host file transfer programs:

System	IBM Program Product Number
CICS/VS	5798-DQH
MVS/TSO	5665-311
VM/CMS	5664-281

SEND and RECEIVE are disk/diskette resident programs that the user invokes as commands from the DOS command level. At the DOS prompt, the user enters SEND or RECEIVE and the appropriate parameters. DOS invokes the SEND or RECEIVE program and the file transfer begins.

You can also invoke SEND and RECEIVE from within a program. To do this, you can use the DOS EXEC function (function number 4BH), or an equivalent function in a high level language if the language provides it.

File transfer supports all controller disks, and controller diskettes with 256-byte sectors. They must be standard labeled diskettes and contain a SYSDSLBL data set. For more information on controller diskettes and data sets, see *4700 Communication System, Controller Programming Library, Volume 2: Disk and Diskette Programming.*

Note: SEND and RECEIVE do not work unless the CCC.COM code is loaded. See "Completing the Operational Disk or Diskette" on page 3-6 for details on loading CCC.COM.

SEND Command

SEND is the command you enter in a DOS session to transfer a DOS file to a host system or 4700 controller.

4700 SEND Command

The format of the 4700 SEND command is as follows:

[a:]SEND [d:][path]filename[.ext] data-set-name [filetype] [drive] [options]

a: Personal computer drive where the disk/diskette with the file transfer command resides. Required only if it is not the default drive.

SEND Required command name that sends a file from the personal computer to the 4700 controller.

d: Personal computer drive where the disk/diskette with the DOS file to be sent resides. If you do not specify this, the system assumes the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be sent resides.

filename Required positional parameter, DOS filename of file to be sent.

.ext DOS file extension. Required if file being sent has an extension.

There should be no spaces between disk drive, path, filename and extension. Refer to the *IBM Personal Computer Disk Operating System* manual for further information on these parameters.

data-set-name

Required positional parameter; the 4700 data set name. If a 4700 data set already exists with the data set name and filetype and on the drive specified in the command, the new data set replaces the old data set. Also, new data set characteristics are the same as those of the old data set unless otherwise specified in the options list.

Note: The personal computer application does not translate lower case characters to upper case; therefore, the data set name must be specified by the user exactly as it is defined at the 4700.

filetype Optional positional parameter; the EDAM file type. The default is ASDS.

Support is provided by 4700 file transfer for the transfer of the following EDAM data set types:

ESDS - EDAM sequential data set - sequential and includes exchange data sets on diskette only; has fixed length records up to sector size.

EDDS - EDAM direct data set - allows direct access through relative record numbers; has fixed length records up to sector size.

ASDS - Arrival sequence data set - direct data set that provides either fixed or variable length records and allows record lengths of up to 1024 bytes.

SEND supports the following record attributes for EDAM files:

ESDS - Blocked
EDDS - Blocked
ASDS - Blocked and spanned.

Transfer of the following EDAM data set types is not supported:

TEMP - Temporary file data set
RKAP - Random keyed access path
KSAP - Keyed sequence access path.

drive Optional positional parameter to indicate 4700 drive. If omitted, the default is A.

1 - Diskette drive 1
2 - Diskette drive 2
A - Disk drive A
B - Disk drive B
C - Disk drive C
D - Disk drive D.

Note: For the 4701, only diskettes formatted with 256 byte sectors are supported. Diskettes must be standard labeled diskettes as described in the *IBM Diskette General Information Manual* and must contain a SYSDSLBL data set as created with the file utility package or the user data set option of the host transmission facility.

For the 4702, diskettes must be 5 1/4 in., high capacity, and must be formatted and initialized by the File Utility Package.

Note: A space is required in the SEND command line between the last positional parameter and the first keyword parameter.

The following keyword parameters are optional:

/C CRLF - Specifies whether carriage return/line feed characters are recognized as record separators and deleted before storing to the 4700 storage medium.

/D DCOMP - Specifies that diskette compress is to be performed when the /M option is specified and the SYSAP data set is not large enough to contain the new member.

/F FINANCE CONTROLLER - Specifies that the file transfer is to the 4700 Finance Communication System controller. /F is the default.

/K KEYSTROKES - Specifies that the IPTXFER command is transmitted to the 4700 as keystrokes (LDA 0 input). If you omit this operand, the IPTXFER command is sent in a data frame (LDA 7 input).

/L:n LRECL n - Specifies the logical record length of the 4700 data set records where n is the number of characters in each record. In the case of RECFM V, this is the maximum size record expected.

If an existing 4700 data set is being replaced, its record length is used as the record length of the new version of the replaced data set except if LRECL is provided in the parameter list. When LRECL is provided in the parameter list for a data set that is being replaced, the parameter list value supersedes the former record length of the data set.

With the exception of the replace condition described, if LRECL is omitted, the default value is 80.

/M:name MEMBER - Specifies the name of a member of the SYSAP data set. The personal computer file with the specified name is added to the SYSAP data set as a member with the specified member name. If a member with that name already exists, it is replaced. This parameter must be specified if the name of the SYSAP data set was specified for the data set name, or the entire SYSAP data set will be replaced.

The SYSAP data set must be large enough to contain the member, or the /D option must be specified.

If uploading a member to a SYSAP on disk, the /S option must be specified, with a size specification of at least the size of the member.

/R:format RECFM - Specifies the record format and characteristics of the 4700 data set.

format is F for fixed length and V for variable length.

For new data sets, if RECFM is omitted, the default is V when the default filetype of ASDS is used or when the ASDS data set type is specified. When ESDS or EDDS filetype is specified, F is the default.

/S:n SIZE n - Required keyword parameter for SEND if the transfer is to the 4700 disk or to the 4700 5 1/4 in. diskette. The 4700 file transfer program ignores this parameter if the transfer is to the 4700 8 in. diskette.

n is the number of k-bytes to be allocated to the 4700 data set.

The user must specify an adequate number of k-bytes to be allocated to the data set for a successful transfer. Consideration must be given to the efficiency of sector utilization with regard to logical record length and record attributes when specifying SIZE as the data set may require more storage at the 4700 than at the personal computer.

/T TRANSLATE - Used when the DOS file is stored in ASCII form and is to be translated to EBCDIC by the 4700 File Transfer program. It causes the 4700 program to translate records from ASCII to EBCDIC prior to writing them to the data set.

This option is only required if the data will be processed by a 4700 application. If the only purpose in sending it is so that another personal computer can receive it, it need not be specified either time.

/X:dssssss...d

EXIT PARAMETER STRING - Literal parameter string to be passed to IPTXIT, the 4700 user-exit application.

d is the delimiter character for the string of characters to be passed. The delimiter characters are transmitted in the command line to IPTXFER but are not passed to IPTXIT. Any valid character that is not in the string can be used as a delimiter, except characters recognized by DOS that redirect standard I/O and chain programs by piping of standard I/O. (Refer to the *IBM Personal Computer Disk Operating System* manual for further information on I/O and piping). SEND recognizes the first character after the colon as the delimiter. SEND then interprets all characters as exit parameter string characters until it encounters the next occurrence of the character it recognized as the delimiter character at the beginning of the string.

sssss... is the user exit parameter string to be passed to IPTXIT. You can specify a maximum of 40 characters in addition to the 2 delimiter characters. You can specify any valid character except characters recognized by DOS that redirect standard I/O and chain programs by piping of standard I/O.

CICS/VS SEND Command

The format of the CICS/VS SEND command is as follows:

[a:]SEND [d:][path]filename[.ext] filename [options]

a: Personal computer drive where the disk/diskette with the file transfer command resides. Required only if it is not the default drive.

SEND Required command name that sends a file from the personal computer to the host.

d: Personal computer drive where the disk/diskette with the DOS file to be sent resides. If not specified, the system will assume the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be sent resides.

filename Required positional parameter, DOS filename of file to be sent.

.ext DOS file extension. Required if file being sent has an extension.

There should be no spaces between disk drive, path, filename and extension. (Refer to the *IBM Personal Computer Disk Operating System* manual for further information on the above parameters.)

filename Required positional parameter. The CICS filename.

Note: A space is required in the SEND command line between the last positional parameter and the first keyword parameter.

The following keyword parameter is required:

/H:C HOST CICS - Specifies that the host is a CICS/VS system and IBM Program Product 5798-DQH is installed and File Transfer is to the host.

The following keyword parameters are optional:

/C CRLF - Specifies whether carriage return/line feed characters are recognized as record separators and deleted before storing to the host storage medium.

/T TRANSLATE - Used when the DOS file is stored in ASCII form and is to be translated to EBCDIC by the host file transfer program. It causes the CICS program to translate records from ASCII to EBCDIC prior to writing them to the data set.

MVS/TSO SEND Command

The format of the MVS/TSO SEND command is as follows:

[a:]SEND [d:][path]filename[.ext] data set name [(member name)][/password] [options]

a: Personal computer diskette drive where the diskette with the file transfer command resides. Required only if the diskette is not in the default drive.

SEND Required command name that sends a file from the personal computer to the host.

d: Personal computer diskette drive where the diskette with the DOS file to be sent resides. If not specified, the system will assume the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be sent resides.

filename Required positional parameter, DOS filename of file to be sent.

.ext DOS file extension. Required if file being sent has an extension.

There should be no spaces between the disk drive, path, filename and extension. (Refer to the *IBM Personal Computer Disk Operating System* manual for further information on the above parameters.)

data set name
Required positional parameter; the MVS/TSO data set name. If a host data set already exists with the data set name

specified in the command, the new data set replaces the old data set unless the /A option is specified.

(member name)
Optional positional parameter; the name of a member in a partitioned data set.

/password Optional positional parameter, required if the data set is password protected.

There should be no spaces between data set name, member name and password.

Single quotes can be typed around the combined data set name, member name, and password to indicate that the user ID is not prefixed.

Note: A space is required in the SEND command line between the last positional parameter and the first keyword parameter.

The following keyword parameter is required:

/H:T HOST MVS/TSO - Specifies that the host is an MVS/TSO system and IBM Program Product 5665-311 is installed and file transfer is to the host.

The following keyword parameters are optional:

/A APPEND - Specifies that a personal computer file is to be attached to the end of the host data set. Default is REPLACE. Cannot be specified for members of a partitioned data set.

/B:n BLKSIZE n - Specifies the block size of the host data set where n is the length in bytes of a data block. If /B is omitted, the default is LRECL for new files. If appending or replacing, /B is ignored.

/C CRLF - Specifies whether carriage return/line feed characters are recognized as record separators and deleted before storing to the host storage medium.

/L:n LRECL n - Specifies the logical record length of the host data set records where n is the number of characters in each record. In the case of RECFM V this is the maximum size record expected.

If an existing host data set is being replaced, its record length is used as the record length of the new version of the replaced data set.

With the exception of the replace condition described, if LRECL is omitted, the default value is 80.

/R:format RECFM - Specifies the record format and characteristics of the host data set.

format is **F** for fixed length, **V** for variable length, and **U** for undefined.

For new data sets, if RECFM is omitted, the default is V when /C is specified and F otherwise.

/S:n[,m] SPACE quantity[,increment] - Specifies the amount of space to be allocated for a new data set. If /S is specified, /U can be specified.

/T TRANSLATE - Used when the DOS file is stored in ASCII form and is to be translated to EBCDIC by the host file transfer program. It causes the host program to translate records from ASCII to EBCDIC prior to writing them to the file.

/U:units SPACE UNITS - Specifies the units used for SPACE quantity and increment. Can be specified only when /S is specified.

units is **A(n)** for avblock(value), **C** for cylinders, and **T** for tracks.

VM/CMS SEND Command

The format of the VM/CMS SEND command is as follows:

[a:]SEND [d:][path]filename[.ext] filename filetype [filemode] [options]

a: Personal computer diskette drive where the diskette with the file transfer command resides. Required only if the diskette is not in the default drive.

SEND Required command name that sends a file from the personal computer to the host.

d: Personal computer diskette drive where the diskette with the DOS file to be sent resides. If not specified, the system will assume the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be sent resides.

filename Required positional parameter, DOS filename of file to be sent.

.ext DOS file extension. Required if file being sent has an extension.

There should be no spaces between disk drive, path, filename and extension. (Refer to the *IBM Personal Computer Disk Operating System* manual for further information on these parameters.)

filename Required positional parameter. The VM/CMS filename. If a host file already exists with the filename, filetype and filemode specified in the command, the

new file replaces the old file unless the /A option is specified.

filetype Required positional parameter. The VM/CMS filetype.

filemode Optional positional parameter to indicate VM/CMS filemode. If omitted, the default is A1.

Note: A space is required in the SEND command line between the last positional parameter and the first keyword parameter.

The following keyword parameter is required:

/H:V HOST VM/CMS - Specifies that the host is a VM/CMS system and IBM Program Product 5664-281 is installed and File Transfer is to the host.

The following keyword parameters are optional:

/A APPEND - Specifies that the personal computer file is to be attached to the end of the host file. Default is REPLACE.

/C CRLF - Specifies whether carriage return/line feed characters are recognized as record separators and deleted before storing to the host storage medium.

/L:n LRECL n - Specifies the logical record length of the host file records where n is the number of characters in each record. In the case of RECFM V, this is the maximum size record expected.

If an existing host file is being replaced, its record length is used as the record length of the new version of the replaced file except if LRECL is provided in the parameter list. When LRECL is provided in the parameter list for a file that is being replaced, the parameter list value supersedes the former record length of the file.

With the exception of the replace condition described, if LRECL is omitted, the default value is 80.

/R:format RECFM - Specifies the record format and characteristics of the host file.

format is F for fixed length and V for variable length.

For new files, if RECFM is omitted, the default is V when /C is specified and F otherwise. During a replace operation, the RECFM of the existing file is the default.

/T TRANSLATE - Used when the DOS file is stored in ASCII form, and is to be translated to EBCDIC by the host file transfer program. It causes the host program to translate records from ASCII to EBCDIC prior to writing them to the file.

Examples Using the SEND Command

This example shows how to use the 4700 SEND command with the following options and conditions:

- The SEND.COM program resides on drive C of the personal computer
- Source DOS file name FIL.ASC on drive A of the personal computer
- Destination 4700 controller data set name NEWFILE
- 4700 filetype ESDS
- Destination 4700 controller disk drive B
- DOS file is stored in ASCII form and is to be converted to EBCDIC by the 4700 File Transfer Program
- Carriage return/Line feed characters are to be removed before storing the 4700 data set
- The amount of storage to be allocated for the NEWFILE data set is 8k bytes.

At the DOS prompt, enter the following line:

```
C:SEND A:FIL.ASC NEWFILE ESDS B
/F/T/C/S:8
```

This example shows how to use the CICS SEND command with the following options and conditions:

- The SEND.COM program resides on drive C of the personal computer
- Source DOS file name FIL.ASC on drive A of the personal computer
- Destination CICS filename NEWFILE
- DOS file is stored in ASCII form and is to be converted to EBCDIC by the CICS File Transfer Program
- Carriage return/Line feed characters are to be removed before storing the CICS file.

At the DOS prompt, enter the following line command line:

```
C:SEND A:FIL.ASC NEWFILE     /H:C/T/C
```

This example shows how to use the MVS/TSO SEND command with the following options and conditions:

- The SEND.COM program resides on drive C of the personal computer
- Source DOS file name FIL.ASC on drive A of the personal computer
- Destination TSO data set name NEWFILE
- NEWFILE data set is a partitioned data set and NEWMBR is a member
- NEWFILE data set is password protected and PW123456 is the password
- DOS file is stored in ASCII form and is to be converted to EBCDIC by the TSO File Transfer Program
- Carriage return/Line feed characters are to be removed before storing the TSO data set
- The logical record length is to be set to 132
- The record format is fixed
- An initial space of 20 tracks is to be allotted in increments of 10.

At the DOS prompt, enter the following line:

```
C:SEND A:FIL.ASC 'NEWFILE(NEWMBR)
/PW123456'    /H:T/T/C/L:132/R:F
/S:20,10/U:T
```

This example shows how to use the VM/CMS SEND command with the following options and conditions:

- The SEND.COM program resides on drive C of the personal computer
- Source DOS file name FIL.ASC on drive A of the personal computer
- Destination CMS filename NEWFILE
- Destination CMS filetype DATA
- Destination CMS filemode A1
- DOS file is stored in ASCII form and is to be converted to EBCDIC by the CMS File Transfer Program
- Carriage return/Line feed characters are to be removed before storing the CMS file
- The record format is variable.

At the DOS prompt, enter the following command line:

```
C:SEND A:FIL.ASC NEWFILE DATA A1
/H:V/T/C/R:V
```

RECEIVE Command

RECEIVE is the command entered in a DOS session to transfer a host or 4700 file to the personal computer.

4700 RECEIVE Command

The format of the 4700 RECEIVE command is as follows:

[a:]RECEIVE [d:][path]filename[.ext] data-set-name [drive] [options]

a: Personal computer drive where the disk/diskette with the file transfer command resides. Required only if it is not the default drive.

RECEIVE Required command name that receives a file from the 4700 to the personal computer.

d: Personal computer drive where the disk/diskette with the DOS file to be received is to reside. If not specified, the system assumes the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be received is to reside.

filename Required positional parameter, DOS filename of file to be received. If a DOS file already exists with the filename and filetype and on the drive specified in the command, the new file replaces the old file.

.ext Personal computer file extension. Required if file being received is to have an extension.

There should be no spaces between disk drive, path, filename and extension. (For more information on these parameters, see the *IBM Personal Computer Disk Operating System* manual.)

data-set-name

Required positional parameter. The 4700 data set name.

Note: The personal computer does not translate lower case characters to upper case; therefore, the data set name must be specified by the user exactly as it is defined at the 4700.

drive Optional positional parameter to indicate 4700 drive. If omitted, the default is A.

1 - Diskette drive 1
2 - Diskette drive 2
A - Disk drive A
B - Disk drive B
C - Disk drive C
D - Disk drive D.

Note: A space is required in the SEND command line between the last positional parameter and the first keyword parameter.

The following keyword parameters are optional:

/C CRLF - Specifies whether carriage return/line feed characters are to be stored as the last two characters of each line when a file is stored on the personal computer storage medium.

/F FINANCE CONTROLLER - Specifies that the file transfer is from the 4700 Finance Communications System controller. /F is the default.

/K KEYSTROKES - Specifies that the IPTXFER command is transmitted to the 4700 as keystrokes (LDA 0 input). If you omit this operand, the IPTXFER command is sent in a data frame (LDA 7 input).

/M:name MEMBER name of member of the SYSAP data set. The member with specified name is sent to the specified PC file. You must specify this option if the name of the SYSAP data set was specified for the data set name, or the entire SYSAP data set will be sent.

/T TRANSLATE - Used when the 4700 data set is stored in EBCDIC form the DOS file is to be stored in ASCII form. It causes the 4700 program to translate records from EBCDIC to ASCII prior to transmitting them to the personal computer.

/X:dssssss...d

EXIT PARAMETER STRING - Literal parameter string to be passed to IPTXIT, 4700 user-exit application.

d is the delimiter character to delimit the string of characters to be passed. The delimiter characters are transmitted in the command line to IPTXFER but are not passed to IPTXIT. Any valid character not in the string can be used as a delimiter, except characters recognized by DOS that redirect standard I/O and chain programs by piping of standard I/O. RECEIVE recognizes the first character after the colon as the delimiter and interprets all characters as exit parameter string characters until it encounters the next occurrence of the character it recognized as the delimiter character at the beginning of the string.

sssss... is the user exit parameter string to be passed to IPTXIT. You can specify a maximum of 40 characters in addition to the 2 delimiter characters. Any valid character may be specified except characters recognized by DOS that redirect standard I/O and chain programs by piping of standard I/O.

See the *IBM Personal Computer Disk Operating System* manual for details on personal computer I/O and piping.

CICS/VS RECEIVE Command

The format of the CICS/VS RECEIVE command is as follows:

[a:]RECEIVE [d:][path]filename[.ext] filename [options]

a: Personal computer drive where the disk/diskette with the file transfer command resides. Required only if it is not in the default drive.

RECEIVE Required command name that receives a file from the host to the personal computer.

d: Personal computer drive where the disk/diskette with the DOS file to be received is to reside. If not specified, the system assumes the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be received is to reside.

filename Required positional parameter, DOS filename of file to be received. If a DOS file already exists with the filename and filetype and on the drive specified in the command, the new file replaces the old file.

.ext Personal computer file extension. Required if file being received is to have an extension.

There should be no spaces between the disk drive, path, filename and extension. (For more information on these parameters, refer to the *IBM Personal Computer Disk Operating System* manual.)

filename Required positional parameter. The CICS filename.

Note: A space is required in the RECEIVE command line between the last positional parameter and the first keyword parameter.

The following keyword parameter is required:

/H:C HOST CICS - Specifies that the host is a CICS/VS system and IBM Program Product 5798-DQH is installed and file transfer is from the host.

The following keyword parameters are optional:

/C CRLF - Specifies whether carriage return/line feed characters are to be stored as the last two characters of each line when a file is stored on the personal computer storage medium.

/T TRANSLATE - Used when the CICS file is stored in EBCDIC form and the DOS file is to be stored in ASCII form. It causes the host program to translate records from EBCDIC to ASCII prior to transmitting them to the personal computer.

MVS/TSO RECEIVE Command

The format of the MVS/TSO RECEIVE command is as follows:

[a:]RECEIVE [d:][path]filename[.ext] data set name [(member name)][/password] [options]

a: Personal computer drive where the disk/diskette with the file transfer command resides. Required only if it is not in the default drive.

RECEIVE Required command name that receives a file from the host to the personal computer.

d: Personal computer drive where the disk/diskette with the DOS file to be received will reside. If not specified, the system assumes the default drive.

path Optional DOS directory specifier for the directory where the DOS file to be received will reside.

filename Required positional parameter, DOS filename of file to be received. If a DOS file already exists with the filename and filetype and on the drive specified in the command, the new file replaces the old file.

.ext Personal computer file extension. Required if file being received is to have an extension.

There should be no spaces between the disk drive, path, filename and extension. (For further information on these parameters, refer to the *IBM Personal Computer Disk Operating System* manual.)

data set name
Required positional parameter. The MVS/TSO data set name.

(member name)
Optional positional parameter. The name of a member in a partitioned data set.

/password Optional positional parameter, required if the data set is password protected.

There should be no spaces between the data set name, member name, and password.

You can type single quotes around the combined data set name, member name, and password to indicate that the user ID is not prefixed.

Note: A space is required in the RECEIVE command line between the last positional parameter and the first keyword parameter.

The following keyword parameter is required:

/H:T HOST MVS/TSO - Specifies that the host is an MVS/TSO system and IBM Program Product 5665-311 is installed and file transfer is from the host.

The following keyword parameters are optional:

/C CRLF - Specifies whether carriage return/line feed characters are to be stored as the last two characters of each line when a file is stored on the personal computer storage medium.

/T TRANSLATE - Used when the TSO data set is to be stored in EBCDIC form and the DOS file is to be stored in ASCII form. It causes the host program to translate records from EBCDIC to ASCII prior to transmitting them to the personal computer.

VM/CMS RECEIVE Command

The format of the VM/CMS RECEIVE command is as follows:

[a:]RECEIVE [d:][path]filename[.ext] filename filetype [filemode] [options]

a:	Personal computer drive where the disk/diskette with the file transfer command resides. Required only if it is not the default drive.
RECEIVE	Required command name that receives a file from the host to the personal computer.
d:	Personal computer drive where the disk/diskette with the DOS file to be received is to reside. If not specified, the system assumes the default drive.
path	Optional DOS directory specifier for the directory where the file to be received will reside.
filename	Required positional parameter, DOS filename of file to be received. If a DOS file already exists with the filename and filetype and on the drive specified in the command, the new file replaces the old file.
.ext	Personal computer file extension. Required if file being received is to have an extension.

There should be no spaces between the disk drive, path, filename and extension. (For further information on these parameters, refer to the *IBM Personal Computer Disk Operating System* manual.)

filename Required positional parameter. The VM/CMS filename.

filetype Required positional parameter. The VM/CMS filetype.

filemode Optional positional parameter to indicate VM/CMS filemode. If omitted, the default is A1.

Note: A space is required in the RECEIVE command line between the last positional parameter and the first keyword parameter.

The following keyword parameter is required:

/H:V HOST VM/CMS - Specifies that the host is a VM/CMS system and IBM Program Product 5664-281 is installed and file transfer is from the host.

The following keyword parameters are optional:

/C CRLF - Specifies whether carriage return/line feed characters are to be stored as the last two characters of each line when a file is stored on the personal computer storage medium.

/T TRANSLATE - Used when the CMS file is stored in EBCDIC form and the DOS file is to be stored in ASCII form. It causes the host program to translate records from EBCDIC to ASCII prior to transmitting them to the personal computer.

Examples Using the RECEIVE Command

This example shows how to use the 4700 RECEIVE command with the following options and conditions:

- The RECEIVE.COM program resides on drive C of the personal computer
- Destination DOS file name FIL.ASC on drive A of the personal computer
- Source 4700 controller data set name OLDFILE
- Source 4700 controller disk drive B
- 4700 data set is stored in EBCDIC form and is to be converted to ASCII by the the 4700 File Transfer Program
- Carriage return/Line feed characters are to be added before transferring the file.

At the DOS prompt, enter the following line:

```
C:RECEIVE A:FIL.ASC OLDFILE B    /F/T/C
```

This example shows how to use the CICS RECEIVE command with the following options and conditions:

- The RECEIVE.COM program resides on drive C of the personal computer
- Destination DOS file name FIL.ASC on drive A of the personal computer
- Source CICS file name OLDFILE
- CICS file is stored in EBCDIC form and is to be converted to ASCII by the CICS File Transfer Program
- Carriage return/Line feed characters are to be added before transferring the file.

At the DOS prompt, enter the following line:

```
C:RECEIVE A:FIL.ASC OLDFILE    /H:C/T/C
```

This example shows how to use the MVS/TSO RECEIVE command with the following options and conditions:

- The RECEIVE.COM program resides on drive C of the personal computer
- Destination DOS file name FIL.ASC on drive A of the personal computer
- Source TSO data set name OLDFILE
- OLDFILE data set is a partitioned data set and OLDMBR is a member
- OLDFILE data set is password protected and PW123456 is the password
- TSO data set is stored in EBCDIC form and is to be converted to ASCII by the TSO File Transfer Program
- Carriage return/Line feed characters are to be added before transferring the file.

At the DOS prompt, enter the following command line:

```
C:RECEIVE A:FIL.ASC
'OLDFILE(OLDMBR)/PW123456'    /H:T/T/C
```

This example shows how to use the VM/CMS RECEIVE command with the following options and conditions:

- The RECEIVE.COM program resides on drive C of the personal computer
- Destination DOS file name FIL.ASC on drive A of the personal computer
- Source CMS filename OLDFILE
- Source CMS filetype DATA
- Source CMS filemode A1
- CMS file is stored in EBCDIC form and is to be converted to ASCII by the CMS File Transfer Program
- Carriage return/Line feed characters are to be added before transferring the file.

At the DOS prompt, enter the following line:

```
C:RECEIVE A:FIL.ASC OLDFILE DATA A1
  /H:V/T/C
```

SEND and RECEIVE Messages

SEND.COM and RECEIVE.COM produce messages:

- To advise the operator of the status of the file transfer
- When errors are detected in the command syntax
- When I/O errors occur to disk or diskette
- When communications errors occur.

All messages are contained in a file named IPTXFER.MSG that must be available in the current directory on the default drive when SEND.COM or RECEIVE.COM is executed. The Application Services diskette contains files named XFERxxx.MSG where 'xxx' is a number identifying the language in which the message text appears. You select one of these files during customization; it is renamed IPTXFER.MSG for inclusion on the operational diskette.

All messages are written to the standard output device. This is normally the display screen unless redirected by the user when SEND or RECEIVE is issued.

Each message begins with a 4-digit message number and the name of the module issuing the message. This information serves to identify the source of the message in the event that the command is issued from a program or from a BAT file with ECHO OFF.

If the command terminates because of an error or a user option, the number of the last message displayed is used as a return code. You can interrogate return codes by the BATCH subcommands IF and ERRORLEVEL, and by the DOS function call 4DH if SEND.COM or RECEIVE.COM was invoked from a program using DOS function call 4BH.

If the user presses CTRL-BREAK to end SEND.COM or RECEIVE.COM, the return code is 1AH.

In the event that the message file is unavailable or unreadable when SEND.COM or RECEIVE.COM is invoked, or if the message to be displayed is not in the message file, the following message is displayed:

9999 IPTXFER.MSG

Should this problem occur, SEND.COM and RECEIVE.COM will continue to run. However, when a message is to be displayed, only the message number will be output. The list on the following pages can be consulted to determine the missing message text.

Message	Text
0001	SEND.COM: No host/4700 filespec specified.
0001	RECEIVE.COM: No host/4700 filespec specified.

Cause: No host or 4700 file specification was entered on the SEND or RECEIVE command line.

Action: Re-enter valid command with host or 4700 file specification.

Message	Text
0002	SEND.COM: Invalid host/4700 filename specified.
0002	RECEIVE.COM: Invalid host/4700 filename specified.

Cause: The host or 4700 filename or dataset name specified in the SEND or RECEIVE command line has an invalid length or contains an invalid character.

Action: Re-enter command with valid host or 4700 filename or dataset name.

Message	Text
0003	SEND.COM: Invalid host/4700 filetype specified.
0003	RECEIVE.COM: Invalid host/4700 filetype specified.

Cause: The host or 4700 filetype specified in the SEND or RECEIVE command line has an invalid length or contains an invalid character.

Action: Re-enter command with valid host or 4700 filetype.

Message	Text
0004	SEND.COM: Invalid host filemode or 4700 drive specified.
0004	RECEIVE.COM: Invalid host filemode or 4700 drive specified.

Cause: The host filemode or 4700 drive specified in the SEND or RECEIVE command line has an invalid length or contains an invalid character.

Action: Re-enter command with valid host filemode or 4700 drive.

Message	Text
0005	SEND.COM: No personal computer filespec specified.
0005	RECEIVE.COM: No personal computer filespec specified.

Cause: No personal computer file specification was entered on the SEND or RECEIVE command line.

Action: Re-enter valid command with personal computer file specification.

Message	Text
0006	SEND.COM: Invalid personal computer drive specified.
0006	RECEIVE.COM: Invalid personal computer drive specified.

Cause: The personal computer drive specified in the SEND or RECEIVE command line does not exist or contains an invalid character.

Action: Re-enter command with valid personal computer drive.

Message	Text
0007	SEND.COM: personal computer file not found.

Cause: The personal computer file specified in the SEND command line does not exist in the current directory on the specified drive.

Action: Re-enter valid command with correct personal computer file specification.

Message	Text
0008	SEND.COM: personal computer directory not found.
0008	RECEIVE.COM: personal computer directory not found.

Cause: The personal computer path specified in the SEND or RECEIVE command line does not exist on the specified drive.

Action: Re-enter valid command with correct personal computer path specification.

Message	Text
0009	SEND.COM: Unable to access personal computer file.
0009	RECEIVE.COM: Unable to access personal computer file.

Cause: Personal computer file could not be created or opened by SEND or RECEIVE.

Action: Check personal computer diskette, personal computer disk/diskette drive, and/or personal computer file attributes.

Message	Text
0010	SEND.COM: Invalid option specified.
0010	RECEIVE.COM: Invalid option specified.

Cause: An invalid option was entered on the SEND or RECEIVE command line and/or in the DOS command processor's environment.

Action:

- Re-enter command with valid option
- Check options in the environment with the DOS SET command.

Message	Text
0011	SEND.COM: Conflicting options specified.
0011	RECEIVE.COM: Conflicting options specified.

Cause: Mutually exclusive options were entered on the SEND or RECEIVE command line and/or in the DOS command processor's environment.

Action:

- Re-enter command with valid options
- Check the options in the environment with the DOS SET command.

Message	Text
0012	SEND.COM: Unable to transmit command line to host/4700.
0012	RECEIVE.COM: Unable to transmit command line to host/4700.

Cause: The personal computer was unable to transmit command line keystrokes to the host or 4700 controller.

Action: Check personal computer to host/4700 connection.

Note: If the problem still exists after the connection is verified, or if the debugging aid BTRACE.SYS is in use, dump the trace log to a diskette file using the DTRACE.COM module. Then give the diskette containing the dump to your IBM representative.

Message	Text
0013	SEND.COM: Unable to open data transfer path with host/4700.
0013	RECEIVE.COM: Unable to open data transfer path with host/4700.

Cause: The personal computer was unable to establish a connection with the host or 4700 controller.

Action: Check personal computer to host/4700 physical connection.

Note: If the problem still exists after the connection is verified and the debugging aid BTRACE.SYS is in use, dump the trace log to a diskette file using the DTRACE.COM module. Then give the diskette containing the dump to your IBM representative.

Message	Text
0014	RECEIVE.COM: Unable to receive data, error or timeout occurred.

Cause:

- The personal computer timed out while expecting data from the host or 4700 controller, or
- An error occurred while receiving data.

Action:

- Check personal computer to host/4700 connection.
- Check availability of host/4700 file transfer program.
- Check return code from 4700 file transfer program or error message on host session screen.

Note: If you cannot resolve the problem and the debugging aid BTRACE.SYS is in use, dump the trace log to a diskette file using the DTRACE.COM module, and give the diskette containing the dump to your IBM representative.

Message	Text
0015	SEND.COM: Unable to transmit data, error or timeout occurred.

Cause:

- The personal computer timed out while expecting an acknowledgement from the host or 4700 controller, or
- An error notification was received from the host.

Action:

- Check personal computer to host/4700 connection.
- Check availability of host/4700 file transfer program.
- Check return code from 4700 file transfer program or error message on host session screen.

Note: If you cannot resolve the problem and the debugging aid BTRACE.SYS is in use, dump the trace log to a diskette file using the DTRACE.COM module. Give the diskette containing the dump to your IBM representative.

Message	Text
0016	RECEIVE.COM: Personal computer disk/diskette full.

Cause: Personal computer disk/diskette is full.

Action: Make personal computer disk/diskette space available.

Message	Text
0017	SEND.COM: CCC.COM not loaded.
0017	RECEIVE.COM: CCC.COM not loaded.

Cause: The CCC.COM program is not running in the personal computer.

Action: Load CCC.COM.

Message	Text
0018	RECEIVE.COM: Empty file received, no personal computer file created.

Cause: A file with a data length of zero was received.

Action: Check host/4700 file.

Message	Text
0019	SEND.COM: Sending...

Cause: The personal computer is transmitting data to the host/4700.

Action: The message is for information only.

Message	Text
0020	RECEIVE.COM: Receiving...

Cause: The personal computer is receiving data from the host/4700.

Action: The message is for information only.

Message	Text
0021	SEND.COM: Transmitting command to 4700, wait...
0021	RECEIVE.COM: Transmitting command to 4700, wait...

Cause: The personal computer is transmitting the file transfer command to the 4700 controller.

Action: The message is for information only.

Message	Text
0022	SEND.COM: Transmitting command to host, wait...
0022	RECEIVE.COM: Transmitting command to host, wait...

Cause: The personal computer is transmitting file transfer command keystrokes to the host.

Action: The message is for information only.

Message	Text
0023	SEND.COM: Transfer completed.
0023	RECEIVE.COM: Transfer completed.

Cause: The file has been transferred successfully.

Action: None.

Message	Text
0024	SEND.COM: Invalid or unexpected command line parameter.
0024	RECEIVE.COM: Invalid or unexpected command line parameter.

Cause: An invalid positional parameter was encountered in the SEND or RECEIVE command line.

Action: Re-enter valid command.

Message	Text
0025	SEND.COM: Host error.
0025	RECEIVE.COM: Host error.

Cause: An error occurred at the host.

Action: Check error message on host session screen.

Setting Default SEND and RECEIVE Options

You can set default options in the DOS command processor's environment by using the DOS 3.0 SET command. At the DOS prompt enter:

SET IPTOPT=[options]

options is one or more SEND or RECEIVE options in exactly the same form that you would use them with the SEND or RECEIVE command. When you set the options as part of the environment, the SEND and RECEIVE commands read them from the environment.

If you enter an option in the SEND or RECEIVE command line it takes precedence over any inverse option in the environment. For example, /F (4700) in the command line will override /H: (other host) in the environment.

Option operands in the command line also take precedence over different operands for the same option in the environment. For example, /L:132 (LRECL - logical record length 132) in the command line overrides /L:256 (LRECL - logical record length 256) in the environment.

Some options (such as /T - TRANSLATE) do not have operands and also do not have inverse options. If you specify these options in the environment, you cannot override them from the command line.

If a command line option conflicts with an option in the environment, SEND and RECEIVE will terminate with an error message.

Some options are not valid for both SEND and RECEIVE; some are not valid for all SEND commands or for all RECEIVE commands. For example, /L is not valid for any RECEIVE command and /R is valid for the VM/CMS SEND command but not the CICS SEND command.

Before using SEND or RECEIVE, you should remove any option in the environment that will conflict with an option in the SEND or RECEIVE command line. Do this by reissuing the SET command.

For more information on the SET command see the *Disk Operating System* manual for your personal computer.

Section 5. Using Function Calls

Using assembler code to access the Application Services function calls, you can:

- Transfer data to and from the 4700 system
- Simulate operator keystrokes
- Retrieve screen information.

Except for the data transfer and translation functions, the function calls allow your program to simulate a 3278/4704 terminal communicating with the 4700 controller. Functions allow you to read the terminal emulator screen buffer. Thus, your programs can determine responses that the 4700 controller has transmitted to the emulated screen.

Application Services provides the 6FH and 7AH interrupt handlers that are described in the following pages. The Application Services control program CCC.COM interprets these interrupts and calls the communications functions. Be sure that CCC is active; see "Completing the Operational Disk or Diskette" on page 3-6 for details on loading CCC.COM.

Application Services 6Fh Function Calls

Call each of the following functions using the **INT 6FH** interrupt instruction.

Function 8: Present Keystroke

Purpose: Simulates a keystroke from the emulated keyboard to the host 4700 controller.

Format: Call this function by issuing INT 6FH with these register values:

AH = 08H (the function number).
AL = The ASCII code for the keystroke.

Present Keystroke returns the following register value:

AX = 0

Remarks: Function 20, **Issue Keyboard Scan Code**, allows you to send scan codes that this function does not allow. The list below shows the ASCII codes that **Present Keystroke** can send. Refer to the *IBM 4700 Personal Computer Addendum* for the default scan codes sent. Use the Application Services customization program to specify a different ASCII-to-4700 scan-code translation.

Code	Function	Code	Function
00H	Reset	01H	Cursor up
02H	Cursor down	03H	Cursor left
04H	Cursor right	05H	Backtab
06H	Insert mode	07H	Delete character
08H	Backspace	09H	Tab field
0AH	Newline	0BH	Home

Code	Function	Code	Function
0CH	Clear	0DH	Enter
0EH	Erase input	0FH	Erase EOF
10H	(reserved)	11H	PF1
12H	PF2	13H	PF3
14H	PF4	15H	PF5
16H	PF6	17H	PF7
18H	PF8	19H	PF9
1AH	PF10	1BH	PF11
1CH	PF12	1DH	PA1
1EH	PA2	1FH	Test request
20H	Space	21H	!
22H	"	23H	#
24H	$	25H	%
26H	&	27H	'
28H	(	29H	)
2AH	*	2BH	
2CH	,	2DH	-
2EH	.	2FH	/
30H	0	31H	1
32H	2	33H	3
34H	4	35H	5
36H	6	37H	7
38H	8	39H	9
3AH	:	3BH	;
3CH	<	3DH	=

Code	Function	Code	Function
3EH	>	3FH	?
40H	@	41H	A
42H	B	43H	C
44H	D	45H	E
46H	F	47H	G
48H	H	49H	I
4AH	J	4BH	K
4CH	L	4DH	M
4EH	N	4FH	O
50H	P	51H	Q
52H	R	53H	S
54H	T	55H	U
56H	V	57H	W
58H	X	59H	Y
5AH	Z	5BH	[
5CH	U	5DH	]
5EH	Not	5FH	_
60H	‘	61H	a
62H	b	63H	c
64H	d	65H	e
66H	f	67H	g
68H	h	69H	i
6AH	j	6BH	k
6CH	l	6DH	m
6EH	n	6FH	o
70H	p	71H	q

Code	Function	Code	Function
72H	r	73H	s
74H	t	75H	u
76H	v	77H	w
78H	x	79H	y
7AH	z	7BH	Cent
7CH	|	7DH	|
7EH	°	7FH	(reserved)
80H	System request	81H	Device cancel
82H	Alt cursor	83H	Print
84H	Attention	85H	Cursor select
86H	Cursor blink	87H	Print
88H	Clicker	89H	Field mark
8AH	DUP	8BH	Footnote[1]
8CH	Footnote[2]	8DH-92H	(reserved)
93H	PF13	94H	PF14
95H	PF15	96H	PF16
97H	PF17	98H	PF18
99H	PF19	9AH	PF20
9BH	PF21	9CH	PF22
9DH	PF23	9EH	PF24
9FH-FFH	(reserved)		

---

1 Double right (personal computer keyboard)
Reserved (banking keyboards)

2 Double left (personal computer keyboard)
Reserved (banking keyboards)

Function 9: Open Data Transfer Path

Purpose: Opens a data transfer path between the personal computer and the 4700 system.

Format: Call the function by issuing INT 6FH with these register values:

AH = 09H (the function number).

AL = A bit mask with these options:

00000001 = Allow ***reception*** from the 4700 controller.

00000010 = Allow ***transmission*** to the 4700 controller.

ES:BX = Address (segment:offset) of the application's receive data buffer.

CX = Size (in bytes) of the receive buffer. If CX = 0 an internal buffer is allocated (1910 bytes). Use CX = 0 any time a high level language does not allocate buffer space in a fixed location (for example, BASICA).

Open Data Transfer Path returns the following register values:

AX = 0 indicates success.

1 ERROR - path already open.

2 ERROR - segment overflow.

Remarks: You can open a data transfer path for transmitting, receiving, or both.

Error return codes indicate no change to the environment in effect prior to the open request.

You should set the data buffer size to 1910 bytes. This size matches the maximum data-transfer block size. Smaller sizes are permitted, but if you set a smaller buffer size and the data-transfer block size exceeds it, the CCC program discards the extra bytes and presents a buffer overrun error as a response to **Read Data Block**. If you are certain that the 4700 system will never exceed a certain block size, you can set the buffer size to that value.

Function 10: Read Data Block

Purpose: Reads a data block from the work buffer defined in function 9, **Open Data Transfer Path**, to a data buffer in your program.

Format: Call the function by issuing INT 6FH with these register values:

AH = 0AH (the function number).
ES:BX = The address (segment:offset) of your application program's data buffer for this block.
CX = The size of the data buffer in bytes.

Read Data Block returns the following register values:

AX =

0 if successful.
1 if the data transfer path is not open to receive or timeout.
2 if buffer overrun occurred causing lost data.
3 if segment overflow.

CX = The size of the received block in bytes.

Remarks: This function does not return to the calling program until it reads an entire block of data or until a timeout occurs.

Function 11: Test for Data Block Received

Purpose: Tests the receive work buffer to determine whether a complete block of data has been received.

Format: Call the function by issuing INT 6FH with this register value:

AH = 0BH (the function number).

Test for Data Block Received returns the following register values:

AX =

0 if a data block is waiting.
1 if no data block is waiting.
2 if the data block is a message.

CX = The size of the received block. Message status if AX=2.

Remarks: Use this function to check if the receive buffer has data that is ready to be read. This technique avoids loss of control by the calling program if a data block has not been completely received by the personal computer. If a data block is ready, you can read it without delay using function 10, **Read Data Block**.

Function 12: Write Data Block

Purpose: Transmits a block of data to the controller through the data transfer path.

Format: Call this function by issuing INT 6FH with these register values:

AH = 0CH (the function number).
ES:BX = The address (segment:offset) of the data block to send.
CX = The size (in bytes) of the data block to send.

Write Data Block returns one of the following register values:

AX =

0 if successful.
1 if unable to transmit (timeout).
2 if the data transfer path is not open for transmission.
3 if data length greater than 1910.
4 if segment overflow.
5 if receive data is waiting, no data written.
6 retry received from host, abort sent to host.

Function 13: Close Data Stream

Purpose: Terminates any open data transfer path.

Format: Call this function by issuing INT 6FH with this register value:

AH = 0DH (the function number).

Close Data Stream returns this register value:

AX = 0

Remarks: Call this function before exiting or terminating your program.

Function 14: Establish Event Interrupt Handler

Purpose: Sets up an interrupt that transfers program control to a user-written routine after certain Application Services related events (block received or emulator screen modified).

Format: Call this function by issuing INT 6FH with these register values:

AH = 0EH (the function number).
ES:BX = The address (segment:offset) of the interrupt routine.
CX = The interrupting event mask:

0 = cancel event.
1 = block received.
2 = emulator screen modified.

Establish Event Interrupt Handler returns these register values:

AX = 0
ES:BX = Address (segment:offset) of the interrupt routine in effect when this function call was issued.
CX = The previous interrupt routine's interrupt event mask.

Remarks: Your interrupt-handler routine can read the AX register to determine the cause of the interrupt. The bits match the event mask definition for register CX above. You must save and restore any registers that the routine uses and return from the routine using "IRET".

Your event interrupt handler should not use DOS, BIOS, 7Ah, or other 6Fh function requests; the results would be unpredictable.

When the emulator screen modified interrupt is presented, there is no guarantee that the controller has completed the buffer modifications. The user program should allow a reasonable time for the screen modification to be completed since the controller updates the screen in sections.

Before exiting your application program and returning to DOS, you must turn off the "Event Interrupt" function: set the interrupting event mask in register CX to all zeros and call the **Establish Event Interrupt Handler** function.

If you use this function to test for an "emulator screen modified" indication, you can then use Function 16 to read the emulator terminal status line after you exit the interrupt handler routine. Columns 25 and 26 of the emulator status line contain the characters "PC" when the 4700 application has a message or response pending for the personal computer application.

The 4700 does not send the data block until the personal computer application uses function call 11 (test for data block) or function call 10 (read a data block) to respond to the "PC" indicator on the status line.

Function 15: Read Emulation Screen Data

Purpose: Allows a user program to read the data in the terminal emulation screen buffer.

Format: Call this function by issuing INT 6FH with these register values:

AH = 0FH (the function number).
ES:BX = The address (segment:offset) of the area to receive the data.
SI = The virtual screen offset of the first byte to read.
CX = The number of bytes to read.

Read Emulation Screen Data returns the following register values:

AX =

0 if successful.
1 for an attempt to read past the end of the screen image.
2 for segment overrun.

Remarks: This function allows your program to read the information that the 4700 controller sends. This information would normally be displayed on the emulator screen. An error occurs if you request data beyond the end of the screen. The function returns ASCII characters and converts attribute bytes to spaces.

Function 16: Read Terminal Status Line

Purpose: Allows a user program to read the contents of the terminal status line (the 25th line of the terminal screen).

Format: Call this function by issuing INT 6FH with the following register values:

AH = 10H (the function number).
ES:BX = The address (segment:offset) of the 80-byte area to receive the data.

Read Terminal Status Line returns these register values:

AX =
0 if successful.
1 if segment overflow.

Remarks: This function copies the terminal emulator status line into your defined buffer. The 4700 controller maintains the status line. The emulation program translates some of the codes to codes that can be displayed on the personal computer screen. Use the DOS *Technical Reference* to determine which character codes are displayed.

Function 17: Obtain Terminal Status

Purpose: Obtains the status of the terminal.

Format: Call this function by issuing INT 6FH with these register values:

AH = 11H (the function number).

Obtain Terminal Status returns the following register values:

AX = Status word:

00000001 = keyboard locked.
00000010 = controller not active.

Remarks: In order to assess the terminal status, this function tests for the presence of status symbols on the status line (line 25) of the emulation screen buffer. The symbols that the personal computer displays are X for "don't key" and 4700 for "controller active".

Function 18: Issue Power on Reset

Purpose: Sends a power-on reset signal to the 4700 controller.

Format: Call this function by issuing INT 6FH with the following register value:

AH = 12H (the function number).

Issue Power On Reset returns the following register value:

AX = 0.

Remarks: This function has the same effect as turning off a 3278/4704 terminal connected to a 4700 controller and turning it on again. If the controller has shut down due to error conditions, use this function to reset the connection.

Function 19: Query Emulated Cursor Position

Purpose: Gives the location of the cursor on the terminal-emulated screen.

Format: Call the function by issuing INT 6FH with the following register value:

AH = 13H (the function number).

Query Emulated Cursor Position returns these register values:

AX = 0
CH = Cursor row (0-24)
CL = Cursor column (0-79).

Remarks: This function returns the cursor position on the emulated display, regardless of whether the personal computer is currently displaying a DOS screen image or an emulator screen image.

Function 20: Issue Keyboard Scan Code

Purpose: Allows application programs to send keyboard codes that are not allowed by function 8, **Present Keystroke.**

Format: Call the function by issuing INT 6FH with these register values:

AH = 14H (the function number).
BL = Scan code to send.
BH = The shift state coded as:

0 = Unshifted scan code.
1 = Upper case shifted scan code.
2 = Alt scan code.

Issue Keyboard Scan Code returns these register values:

AX =

0 if successful.
1 if invalid 4700 scan code

Remarks: Review how the controller interprets keyboard scan codes before using this function. See the *4700 Finance Communication System, Controller Programming Library, Volume 4: Loop and DCA Device Programming* manual for details.

Function 21: Query Version Number

Purpose: Obtains the version number of the CCC.COM program (the Application Services control program). Prior to version 1.01, this function returned 0Bh in BX. Version 1.01 returns 1 in BH and 1 (decimal) in BL.

Format: Call the function by issuing INT 6FH with this register value:

AH = 15H (the function number).

Query Version Number returns these register values:

AX = 0
BH = Major Version (after version 1.01)
BL = Minor Version (after version 1.01)

Function 22: Read Emulation Screen with Attributes

Purpose: Reads the data from the terminal emulation screen buffer including the 3270 attribute information. You must read the entire 1920-byte buffer. The 3270 attribute bytes are replaced by the coded values shown below.

You specify the location of a 1920-byte area where the screen image is placed. All printable characters are converted to their ASCII equivalents. The 3270 attribute bytes are converted to the following codes:

Hex Code	3270 Field Attribute
0C0h	unprotect,alpha, norm, nondetect
0C1h	unprotect,alpha, norm, detect
0C2h	unprotect,alpha,inten, detect
0C3h	unprotect,alpha, no disp, nondetect
0C4h	unprotect,num, norm, nondetect
0C5h	unprotect,num, norm, detect
0C6h	unprotect,num, inten, detect
0C7h	unprotect,num, no disp, nondetect
0C8h	protect,alpha, norm, nondetect
0C9h	protect,alpha, norm, detect
0CAh	protect,alpha, inten, detect
0CBh	protect,alpha, no disp, nondetect
0CCh	protect,num, norm, nondetect
0CDh	protect,num, norm, detect
0CEh	protect,num, inten, detect
0CFh	protect,num, no disp, nondetect

Format: Call the function by issuing INT 6FH with this register value:

AH =	16H (the function number).
ES:BX	The address (segment:offset) of the area to receive the data.

Read Emulation Screen returns this register value:

AX = 0

Remarks: This function differs from function 15:

- With function 22, you must read the entire screen; partial reads are not allowed.
- With this function, attribute information is returned.

Function 255: Query Presence

Purpose: Tests for the presence of the CCC.COM program (the Application Services control program).

Format: Call the function by issuing INT 6FH with this register value:

AH = FFH (the function number).

Query Presence returns these register values:

AX = 0 if CCC.COM is present.
FFxxH if CCC.COM is not present.

Remarks: Be sure to verify that the 6Fh interrupt vector is not 0 before you issue a function call 255.

A 0 returned in AX indicates that CCC.COM is installed (rather than another application that uses the 6Fh interrupt vector).

Application Services 7AH Function Calls

Applications Services offers a subset of the application program interface (API) that is provided by the IBM 3270 Personal Computer Control Program:

Supervisor Services: This service converts service names (called gates) to IDs usable by Application Services

Name Resolution

Session Information Services: The services in this group obtain session information. They include:

Query Session ID
Query Session Parameters
Query Session Cursor

Keyboard Services: The services in this group control keystroke data between Application Services and your application. Use these services to send keystrokes.

Connect to Keyboard
Disable Input
Write Keystroke
Enable Input
Disconnect from Keyboard

Copy Services: The service in this group copies a string of data between the session's presentation space and a buffer. Copy Services includes this service:

Copy String

Operator Information Area Services: The service in this group obtains the status of the operator information area. Operator Information Area Services includes this service:

Read Operator Information Area Group

Note: Many of the function calls in this section require that you store segment and offset addresses in a parameter lists. By convention, place the high-order byte first, followed by the low-order byte. For example, if you are to store the segment address in bytes 4 and 5, and the offset in bytes 6 and 7: store the high order byte of the segment address in byte 4 and the low-order byte in byte 5; store the high-order byte of the offset in byte 6 and the low-order byte in byte 7.

Supervisor Service Requests

This section describes the supervisor-service function calls. Use the service in this section to obtain a gate ID for use in other services.

A *gate* is a group of services that perform a common function. Each gate is assigned a name when the gate is created. Application services provides these gates:

Name	Function
SESSMGR	Session Information Services
KEYBOARD	Keyboard Services
COPY	Copy Services
OIAM	Operator Information Area Services

Application Services refers to each gate with a 16-bit gate ID. Before you can use the services of a gate, obtain the gate ID using the Name Resolution service, function call 81h. Then, save the gate ID for use when you request the services of that gate.

Function 81: Name Resolution

Purpose: Converts the service's "gate" name to its numeric ID.

Format: Call the function by issuing INT 7AH with these register values:

AH = 81H (the function number).
ES:DI = Segment:Offset address of the parameter list

The parameter list must be an 8-byte area that contains the gate name in ASCII characters, padded to the right with blanks.

This function returns the following register values:

BH = 07h
CH = 12h
CL = System Return Code

00h	Successful completion
2Eh	Invalid gate name
2Fh	Invalid AH register specified

DX = Gate ID

Remarks: You need only use this function once for each gate name used in your program. Further request for the same gate will return the same ID.

Session Information Services

This section describes the function calls for the session information services that obtain the session ID, session characteristics, and the current cursor position.

First, use function call 81h using SESSMGR as the gate name in the 81h parameter list.

Return Codes: These function calls return two sets of return codes. Register CL returns system return codes. The first byte of the parameter list returns session-information services return codes.

Function 01: Query Session ID

Purpose: Obtains the session ID, which is a required parameter on later function calls.

Format: Call the function by issuing INT 7AH with these register values:

AH = 09H (the function number).
AL = 01H
BH = 80H
BL = 20H
CX = 0000H
DX = Gate ID for SESSMGR (resolved value from function 81h)
ES:DI = Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h
CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	func. ID(6Bh)
2	1 byte	option code (01h)	unchanged
3	1 byte	data code (41h)	unchanged
4	1 word	offset name array	unchanged
6	1 word	segment name array	unchanged
8	8 bytes	session long name	unchanged

Session Information Services Return Code:

00h	Successful completion
0Ch	Byte 0 of the parameter list is not 0
12h	The name array length is invalid

Name Array Format:

Offset	Length	On Request	Completion
0	1 byte	length (0Eh)	unchanged
1	1 byte	not used	# matching sessns(1)
2	1 byte	not used	session short name(41h)
3	1 byte	not used	session type(03h)
4	1 byte	not used	session id(1)
5	1 byte	reserved	reserved
6-13	8 bytes	reserved	long session name

Remarks:

- You need only issue this request once. Subsequent requests will receive the same session ID.
- Although bytes 2 and 3 of the parameter list are not checked, set them as indicated.

Function 02: Query Session Parameters

Purpose: Obtains the session Parameters. You will use some of these parameters on a Copy String request.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	02H
BH =	80H
BL =	20H
CX =	0000H
DX =	Gate ID for SESSMGR (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(6Bh)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 byte	not used	session type
5	1 byte	not used	session chars
6	1 byte	not used	rows
7	1 byte	not used	columns
8	1 word	reserved	reserved
10	1 word	reserved	reserved

Session ID: Code 01h

Session Information Services Return Code:

00h	Successful completion
02h	Specified session ID is invalid
0Ch	Byte 0 of the parameter list is not 0

Session Type:

This request always returns 03h, a host session.

Session Characteristics:

This function always returns 00h in the form:

Bit 0	1	2-7
EAB	PSS	reserved

EAB = 0 session has base attribute
PSS = 0 session does not support programmed symbols

Rows: This is the number of rows in the session's presentation space, always 24.

Columns: This is the number of columns in the session's presentation space, always 80.

Remarks: Issue this function only once. Subsequent requests return the same session ID and characteristics.

Use the row and column values to compute the maximum offset ending character for this session with Copy String service:

MAXIMUM OFFSET = (ROWS X COLUMNS) - 1

Function 0B: Query Session Cursor

Purpose: Obtains the session cursor type and the row and column addresses for the session.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	0BH
BH =	80H
BL =	20H
CX =	00FFH
DX =	Gate ID for SESSMGR (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(6Bh)
2	1 byte	session ID(01)	unchanged
3	1 byte	not used	cursor type
4	1 byte	not used	row address
5	1 byte	not used	column address

Session ID:

Code 01h.

Session Information Services Return Code:

00h	Successful completion
02h	Specified session ID is invalid
0Ch	Byte 0 of the parameter list is not 0

Cursor Type:

This request always returns 00h, an underscore.

Row Address:

This is the address in the session's presentation space representing the cursor's row, always in the range 0 to 23.

Column Address:

This is the address in the session's presentation space representing the cursor's column, always in the range 0 to 79.

Keyboard Services

This section describes the function calls for the keyboard services that write keystrokes to the session, and enable and disable operator input from the keyboard.

First, use function call 81h using KEYBOARD as the gate name in the parameter list.

AID Keys: AID keys are those keys that, when pressed during a session, cause immediate host interaction. Possible AID keys are: PF1-PF24, Enter, Clear, SysRq, Attn, PA1, and PA2.

Return Codes: These function calls return two sets of return codes. Register CL returns system return codes. The first byte of the parameter list returns keyboard services return codes.

Function 01: Connect to Keyboard

Purpose: Connects to a session for keyboard service.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	01H
BH =	80H
BL =	20H
CX =	0000H
DX =	Gate ID for KEYBOARD (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(62h)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 word	must be 0	unchanged
6	1 word	must be 0	unchanged
8	1 byte	must be 0	unchanged
9	1 byte	reserved	reserved

Session ID:

Code 01h.

Keyboard Services Return Code:

00h	Successful completion
02h	Invalid session ID
04h	Session busy; cannot connect
0Ch	Byte 0 of the parameter list is not 0

Remarks: Use this service with the Disconnect from Keyboard service to serialize access to the other keyboard and copy services. This helps prevent several applications from using the application interface at the same time. To do this, observe these guidelines:

- Always use the Connect to Keyboard service before you issue any other keyboard or copy-string service.
- Always check the return code from the Connect-to-Keyboard service. If the code indicates that a session is already connected, do not issue any further keyboard or copy-string requests until you issue a Connect-to-Keyboard that produces a 'Successful Completion' return code

Function 02: Disconnect from Keyboard

Purpose: Disconnects from a session when finished using Keyboard Services or Copy Services.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	02H
BH =	80H
BL =	20H
CX =	0000H
DX =	Gate ID for KEYBOARD (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(62h)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 word	must be 0	unchanged

Session ID:

Code 01h

Keyboard Services Return Code:

00h	Successful completion
02h	Invalid session ID
04h	Session not connected
0Ch	Byte 0 of the parameter list is not 0

Remarks: Use this service with the Connect-from-Keyboard service to serialize access to the other keyboard and copy services. This helps prevent several applications from using the application interface at the same time. To do this, observe these guidelines:

- Always use the Disconnect-from-Keyboard service when you finish using the keyboard or copy-string service.
- Before exiting your application, use the Disconnect-from-Keyboard service if the session was connected for keystroke data. Use this service at all error points as well as during normal processing.

Function 04: Write Keystroke

Purpose: Sends keystroke data to the session. This service is complete when the processing finishes as if the operator entered the keystroke data.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	04H
BH =	80H
BL =	20H
CX =	0000H
DX =	Gate ID for KEYBOARD (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Basic Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(62h)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 word	must be 0	unchanged
6	1 byte	options	unchanged
7	1 byte	not used	number keys sent

options: To send one keystroke, code 20h; to send a list of keystrokes, code 30h.

Follow with Single Keystroke or Keystroke List

To Send a Single Keystroke:

Offset	Length	On Request	Completion
8	1 byte	key scancode	unchanged
9	1 byte	key shift state	unchanged

To Send a List of Keystrokes:

Offset	Length	On Request	Completion
8	1 word	offset of list	unchanged
10	1 word	segment of list	unchanged

Keystroke List:

Offset	Length	Contents
0	1 word	2 times number keys to send
2	1 byte	scancode of first key
3	1 byte	shift state of first key
4	1 byte	scan code of second key
5	1 byte	shift state of second key
.		
.		
2n	1 byte	scan code of nth key
2n+1	1 byte	shift state of nth key

Shift State:

Specify a 1-byte value that indicates the function or character, as printed on a keytop, to send. The format of the shift state is:

0,1	2	3	4	5	6	7
reserved	right shift	left shift	control key	alt keys	shift lock	upshift keys

bits 0,1 reserved

2 represents the right shift key

3 represents the left shift key

4 represents the control shift state

5 represents the ALT shift state

6 represents the shift lock state

7 represents the upshift state.

Bit 7 indicates that one of the two upshift keys was pressed. To distinguish between the two, use bits 2 and 3.

scan code byte:

The scan code is a unique hexadecimal value that is assigned to each key recognized by Application Services. The following table shows the scan codes recognized.

ScanCode	Unshifted	Upshifted	Alternate
03	invalid	invalid	cursor select
05	invalid	invalid	system request
06	clear	invalid	test request
07	PF1	invalid	invalid
08	PF13	invalid	invalid
0B	erase EOF	invalid	erase input
0C	invalid	invalid	attention
0D	tab field	invalid	invalid
0F	PF2	invalid	invalid
10	PF14	invalid	invalid
15	q	Q	invalid
16	1	!	invalid
17	PF3	invalid	invalid
18	PF8	invalid	invalid
1A	z	Z	invalid
1B	s	S	invalid
1C	a	A	invalid
1D	w	W	invalid
1E	2	@	invalid
1F	PF4	invalid	invalid
20	PF16	invalid	invalid
21	c	C	invalid
22	x	X	invalid
23	d	D	invalid
24	e	E	invalid
25	4	$	invalid
26	3	#	invalid
27	PF5	invalid	invalid
28	PF17	invalid	invalid
29	space	invalid	invalid
2A	v	V	invalid
2B	f	F	invalid

ScanCode	Unshifted	Upshifted	Alternate
2C	t	T	invalid
2D	r	R	invalid
2E	5	%	invalid
2F	PF6	invalid	invalid
30	PF18	invalid	invalid
31	n	N	invalid
32	b	B	invalid
33	h	H	invalid
34	g	G	invalid
35	y	Y	invalid
36	6	invalid	invalid
37	PF7	invalid	invalid
38	PF19	invalid	invalid
3A	m	M	invalid
3B	j	J	invalid
3C	u	U	invalid
3D	7	&	invalid
3E	8	*	invalid
3F	PF8	invalid	invalid
40	PF20	invalid	invalid
41	,	<	invalid
42	k	K	invalid
43	i	I	invalid
44	o	O	invalid
45	0	)	invalid
46	9	(	invalid
47	PF9	invalid	invalid
48	PF21	invalid	invalid
49	.	>	invalid
4A	/	?	invalid
4B	l	L	invalid
4C	;	:	invalid
4D	p	P	invalid
4E	-	_	invalid
4F	PF10	invalid	invalid
50	PF22	invalid	invalid
52	'	"	invalid
53	]	invalid	invalid
54	[	invalid	invalid

ScanCode	Unshifted	Upshifted	Alternate
55	=	invalid	invalid
56	PF11	invalid	invalid
57	PF23	invalid	ALT cursor
58	enter	invalid	invalid
5A	newline	invalid	invalid
5E	PF12	invalid	invalid
5F	PF24	invalid	invalid
60	cursor down	invalid	invalid
61	cursor left	invalid	invalid
63	cursor up	invalid	invalid
64	backtab	invalid	invalid
65	insert	invalid	invalid
66	backspace	invalid	invalid
67	PA1	DUP	invalid
69	invalid	1	invalid
6A	cursor right	invalid	invalid
6B	cursor left	4	invalid
6C	home	7	invalid
6D	delete char	invalid	invalid
6E	PA2	field mark	invalid
70	insert	0	invalid
71	delete char	invalid	invalid
72	cursor down	2	invalid
73	invalid	5	invalid
74	cursor right	6	invalid
75	cursor up	8	invalid
79	enter	invalid	invalid
7A	invalid	3	invalid
7D	invalid	9	invalid
83	invalid	print	invalid

Number of Keystrokes Sent:

This is the number of keystrokes sent to the session, up to and including the first keystroke that generated an AID, or to the first rejected keystroke. A keystroke is rejected if it is not a valid scan code or if it cannot be processed by the session due to an input-inhibited condition. Once an AID-generating or invalid keystroke is received, no further keystrokes are processed from the list.

Keyboard Services Return Code:

00h	Successful completion
01h	Invalid options byte
02h	Invalid session ID, or the list of keystrokes is longer than 255 bytes
04h	Session not connected for keyboard services
0Ch	Byte 0 of the parameter list is not 0
10h	Invalid scan code or input inhibited; processing stopped
12h	Successful completion; AID key generated

Remarks:

- You can send up to 255 keystrokes in a list.
- Do not imbed AID keys in a keystroke list. (However you can put an AID key as the last key in a list.) An AID key or an invalid key stops processing of the list.
- If you receive a 10h or 12h return code, check byte 7 of the returned parameter list to find the number of keystrokes actually sent.
- If the value in the first word in the parameter list is not a multiple of 2, application services rounds that value *DOWN* to the next multiple of 2. It is treated as 1 less keystroke.

Function 05: Disable Input

Purpose: Disables operator input to the session.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	05H
BH =	80H
BL =	20H
CX =	0000H
DX =	Gate ID for KEYBOARD (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(62h)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 word	must be 0	unchanged

Session ID:

Code 01h

Keyboard Services Return Code:

00h	Successful completion
02h	Invalid session ID
04h	Session not connected for keyboard services
0Ch	Byte 0 of the parameter list is not 0

Remarks: Keystrokes typed in from a terminal can become mixed in with the keystroke data and copy-string data that your program sends. To prevent this, use this service to diasble the processing of terminal operator input.

Function 06: Enable Input

Purpose: Enables operator input to the session.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	06H
BH =	80H
BL =	20H
CX =	0000H
DX =	Gate ID for KEYBOARD (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(62h)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 word	must be 0	unchanged

Session ID:

Code 01h

Keyboard Services Return Code:

00h	Successful completion
02h	Invalid session ID
04h	Session not connected for keyboard services
0Ch	Byte 0 of the parameter list is not 0

Remarks: Use this service to re-enable input from a terminal operator.

The Disconnect from Keyboard service also enables input if input had been disabled.

Copy Services

This section describes the function calls for the copy services that copy strings of data between the session's presentation space and a buffer.

Before you use this service, use the Connect-for-Keyboard service to make serial the access to the keyboard, and use the Disconnect-from-Keyboard service when you are finished.

Then use function 81h with COPY as the gate name in the parameter list.

Return Codes: These function calls return two sets of return codes. Register CL returns system return codes. The first byte of the parameter list returns copy-services return codes.

Function 01: Copy String

Purpose: Copies data from your application's buffer to the presentation space, or from the presentation space to your application's buffer. The service is complete when the data is moved physically.

This service does not change the cursor position.

Format: Call the function by issuing INT 7AH with these register values:

AH = 09H (the function number).
AL = 01H
BH = 80H
BL = 20H
CX = 00FFH
DX = Gate ID for COPY (resolved value from function 81h)
ES:DI = Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h
CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(64h)
2	1 byte	source session ID	unchanged
3	1 byte	reserved	reserved
4	1 word	offset source buff	unchanged
6	1 word	segment source buff	unchanged
8	1 byte	session chars	unchanged
9	1 byte	source session type	unchanged
10	1 word	offset source start	unchanged
12	1 word	offset source end	unchanged
14	1 byte	target session ID	unchanged
15	1 byte	reserved	reserved
16	1 word	offset target buff	unchanged
18	1 word	segment target buff	unchanged
20	1 byte	target chars	unchanged
21	1 byte	target session type	unchanged
22	1 word	offset target start	unchanged
24	1 byte	copy mode	unchanged
25	1 byte	reserved	reserved

Session ID:

- Use query-session ID to get a session ID.
- To copy from the session to a buffer: put the session ID in the source-session ID field; put the offset and segment in the target buffer fields. To copy from a buffer to the session: put the session ID in the target session ID field; put the offset and segment in the source buffer fields.
- Do not specify both a session ID and a buffer address for the source or target. If the session ID is 0, specify the session buffer adddress; if the target ID is 0, specify the target buffer address.
- Copy strings only from a buffer to the session's presentation space, or from the session to the buffer.

Buffer addresses:

The buffer address is an address in your application.

Session characteristics:

The source or target session characteristics are required only for the application buffer. They must be in this format:

Bit 0	1	2-7
EAB	PSS	reserved

bit 0=0 indicates session has base attributes
bit 1=0 indicates session does not support programmed symbols

Session type:

Code a session type only for the application buffer. It determines the data format of the buffer, and must be 03h, a host session.

Offset starting character:

Code the offset into the buffer or presentation space of the starting character of the string. Do not include field attributes. This value must be in the range 0 to 1919 for the presentation space.

Offset source ending character:

Code the character offset into the presentation space or buffer, which ever is the source of the string. Do not include field attributes. This number must be in the range 0 to 1919 if it is in the presentation space.

Copy Mode:

Code this only if the application buffer is the target.

00h	field attributes not copied
40h	field attributes copied.

Copy Service Return Code:

00h	Successful completion
02h	Invalid session ID
03h	Target window is input-inhibited
06h	Missing or invalid source parameter
07h	Missing or invalid target parameter
09h	Data truncated; copy performed
0C	Byte 0 of parameter list not 0
0Eh	Target window is protected
0Fh	Copy of field attributes not allowed (copy not performed)

Remarks:

When the presentation space is the target: If the starting and ending offsets would cause the copy to extend beyond the end of the presentation space, the data is copied to the end of the presentation space. Copy-service return code 09h indicates truncation.

When the presentation space is the source: if the starting or ending character offsets are out of the acceptable range, the copy is not performed. Copy-service return code 06h is returned.

If the starting source offset is greater than the ending source offset, no copy is made. Copy-service return code 06h is returned.

Operator Information Area Services

This section describes the function calls for the operator information area services that obtain the status of the operator information area.

The Read Operator Information Area service returns a bit string that reflects the current settings of indicators in the operator information area.

First use function 81h with OIAM as the gate name in the parameter list.

Return Codes: These function calls return two sets of return codes. Register CL returns system return codes. The first byte of the parameter list returns operator-information service return codes.

Function 01: Read Operator Information Area

Purpose: Returns a bit string that reflects the current settings of Operator Information Area indicators for this session.

Format: Call the function by issuing INT 7AH with these register values:

AH =	09H (the function number).
AL =	02H
BH =	80H
BL =	20H
CX =	00FFH
DX =	Gate ID for OIAM (resolved value from function 81h)
ES:DI =	Segment:Offset address of the parameter list

This function returns the following register values:

CH = 12h

CL = System Return Code

00h	Successful completion
05h	Invalid DX register specified
07h	Invalid BH register specified
08h	Invalid BL register specified
2Fh	Invalid AH register specified
34h	Invalid AL register specified

Parameter List:

Offset	Length	On Request	Completion
0	1 byte	must be 0	return code
1	1 byte	must be 0	function ID(6Dh)
2	1 byte	session ID(01)	unchanged
3	1 byte	reserved	reserved
4	1 word	offset in buffer	unchanged
6	1 word	segment in buffer	unchanged
8	1 byte	must be 08h	unchanged

Session ID:

Code 01h.

Buffer addresses:

Code the segment and offset addresses for the buffer into which the bit string is to be read. This must be a 5-byte buffer.

Group Indicator Meanings:

The returned indicators follow:

0	0	Returned as 0	2	0	Input Inhibited
	1	Returned as 0		1	Input Inhibited
	2	Returned as 0		2	Input Inhibited
	3	Returned as 0		3	Input Inhibited
	4	Returned as 0		4	Input Inhibited
	5	Input Inhibited		5-7	Returned as 0
	6	Input Inhibited			
	7	Input Inhibited	3	0	Input Inhibited
1	0	Input Inhibited		1	Input Inhibited
	1	Input Inhibited		2	Input Inhibited
	2	Input Inhibited		3	Input Inhibited
	3	Input Inhibited		4	Input Inhibited
	4	Input Inhibited		5-7	Returned as 0
	5	Input Inhibited			
	6	Input Inhibited	4	0	Input Inhibited

7	Input Inhibited	1-7	Returned as 0

Operator Information Services Return Code:

00h	Successful completion
02h	Invalid session ID
0C	Byte 0 of parameter list not 0

Translation Function Call

Use the translation function call to translate data strings from ASCII to EBCDIC or from EBCDIC to ASCII. Use **INT 6BH** to call this function. Note that this is not the same interrupt as is used for the application services function calls.

Note: The XLATE.COM code must be resident with DOS for translation function calls to work. See "Completing the Operational Disk or Diskette" on page 3-6 for details on loading XLATE.COM.

Function 05H: Translate ASCII/EBCDIC

Purpose: Translates a data string from ASCII to EBCDIC or from EBCDIC to ASCII.

Format: Call the function by issuing INT 6BH with these register values:

AH = 05H (the function number).
AL = Select translation direction:
- 01H = Convert data located at DS:DX from ASCII to EBCDIC.
- 02H = Convert data located at DS:DX form EBCDIC to ASCII.
- FFH = Is function active query (used to determine if XLATE.COM has been loaded - see return codes).

BX = Select translation table:
- 0000H = Standard ASCII/EBCDIC translation.
- 000NH = Select alternate translation table N. N can range from 1 to 7. Corresponds to the table (or one of the tables) you loaded with XLATE.COM.

DS:DX = The address (segment:offset) of the data string to be translated.
CX = The number of bytes to translate.

Translate ASCII/EBCDIC returns the following register values:

AH = Zero, indicating that the request was handled.

AL = A return code that indicates completion status:
- 00H = Good status.
- 01H = Translation table not loaded.
- 02H = Data check (cannot translate FFH character).
- FFH = Request unknown or XLATE.COM not loaded.

CX = Number of untranslated characters if 02H data check error returned.

Remarks: This function translates data in place. If the data buffer you indicate contains ASCII before translation, it will contain EBCDIC after successful translation (or vice versa). The translation is byte by byte and ordered according to the translate table. In other words, the translation function looks up the current byte in the translation table then substitutes the designated translate byte.

You can configure a translate table sequence to do special things like capitalize all characters or remove all capital letters.

Section 6. Using Virtual Volumes

The 4700 controller can contain 4700 data sets (disk files in the 4700 controller) that consist of DOS file data in DOS format. These data sets are called **virtual volumes**. The name of each virtual volume data set begins with the characters "VVOL." followed by a unique data set identifier. The controller stores this name with all upper case characters.

You can access virtual volumes as if they were real personal computer disks (or diskettes for VVOL.SYSTEM) through the use of remote system reset (remote IPL) and the virtual volume device driver.

At the personal computer, virtual volumes appear to be standard disks or diskettes. Most of the standard DOS disk and diskette functions apply (some exceptions are FORMAT, DISKCOPY, DISKCOMP, and FDISK).

The public volume manager personal computer is attached to a designated port of the 4700 controller. The CPGEN process for the 4700 controller includes the option of assigning public volume manager status to one port on the controller. See the *IBM 4700 Personal Computer Addendum to the 4700 Finance Communication System* for more information on CPGEN. The public volume manager has full access to the virtual volumes in the 4700 controller, and can read, write, or modify them. The rest of the personal computers attached to the controller have less access to the virtual volumes.

Use of virtual volumes revolves around two tasks:

- Creation and management of virtual volumes by the public volume manager
- Use of virtual volumes as alternate disks and diskettes.

Virtual Volume Types

You will use three types of virtual volumes. Their names all begin with the character string "VVOL." (the period precedes the data set identifier). These types are:

1. **Remote Reset Volume** (for remote IPL). It always has the name VVOL.SYSTEM, and its contents are read only in normal use. Only the public volume manager can change this volume.

2. **Public Volume.** It always has the name VVOL.PUBLIC, and its contents are also always read only in normal use. Only the public volume manager can change this volume.

3. **Private volumes.** They can have any name beginning with "VVOL." except for the "VVOL.SYSTEM" and "VVOL.PUBLIC" names. Each private volume can be assigned to a specific personal computer. That personal computer has read and write access.

Remote System Reset Volume

VVOL.SYSTEM is the 4700 data set that contains files for remote system reset (remote IPL). This is a **read only** data set for the personal computers that have access to it, excluding the public volume manager personal computer. VVOL.SYSTEM can reside on any 4700 fixed disk. It is formatted like a personal computer diskette and contains, at a minimum, the DOS system files (the files that enable a personal computer to load DOS). VVOL.SYSTEM can also contain files that a personal computer performing a remote IPL uses, such as AUTOEXEC.BAT, CONFIG.SYS, and personal computer device drivers and DOS applications. The principal use of VVOL.SYSTEM is the remote system reset of a personal computer. The 4700 controller can contain only one VVOL.SYSTEM data set.

You can load applications from VVOL.SYSTEM, but these would normally reside on the public volume. Files on VVOL.SYSTEM are only available to personal computers that perform a remote IPL. Personal computers that do not perform a remote IPL can access only VVOL.PUBLIC and a private volume.

A personal computer that uses remote IPL will access the remote system reset volume as drive A. Any real drives it has will start with drive B. If it has no real drives, it will access the remote system reset volume as drives A and B. A personal computer that does not perform remote IPL does not have access to the remote system reset volume.

If you are acting as public volume manager and you want to create a remote system reset volume, you should first create a DOS diskette with all of the files and programs that you want on the remote system reset volume. Then you can use the VVOL command with the /CS option to copy the diskette contents to the new remote system reset volume. You can test the diskette before creating the new VVOL.SYSTEM by attempting an IPL from the diskette.

Public Volume

VVOL.PUBLIC is the 4700 data set that contains the files and programs available to all the personal computers that are attached to the controller. This is a **read only** data set. It contains files that are available to any personal computer that has virtual volume capability. Only the public volume manager can copy or write to files in the VVOL.PUBLIC data set.

VVOL.PUBLIC can reside on any 4700 fixed disk. It is formatted like a personal computer DOS fixed disk and can contain any DOS system or application files that you may want to receive from the 4700 controller. VVOL.PUBLIC is only necessary if one or more users needs access to its files.

Private Volumes

Each personal computer can control one private volume data set in the 4700 controller at any one time. By using the VVOL command, the user of the personal computer can change to a different private volume at any time (if it is not in use).

The public volume manager uses the VVOL command to create private volumes on any of the controller's drives. The CPGEN process for the 4700 controller can specify the association between the personal computer attached to a given port and a specific private volume data set. If this is done, the user of that personal computer does not have to issue the VVOL command. Even if a private volume is not assigned during CPGEN, the user can assign or change the assignment of his private volume at any time by using the VVOL command.

A personal computer has exclusive read and write access to its own private volume. A private volume data set is formatted like a personal computer DOS fixed disk.

The Public Volume Manager

During controller configuration, one DCA port on the 4700 controller can be designated as belonging to the public volume manager personal computer. For details on controller configuration and port assignment, see the *IBM 4700 Personal Computer Addendum to the 4700 Finance Communication System.* The public volume manager personal computer can perform the following functions with 4700 virtual volumes:

- Write (or copy) to the remote system reset volume (data set name VVOL.SYSTEM), which is ***read only*** for other personal computers that are attached to the controller.
- Write (or copy) to the public volume (data set name VVOL.PUBLIC). The data set contents are ***read only*** for the other personal computers that are attached to the controller.
- Create private virtual volume data sets on the 4700 fixed disk for use as virtual disks.
- Erase unneeded virtual volume data sets from the 4700 fixed disk.
- Rename virtual volume data sets.

The public volume manager personal computer needs at least one real diskette drive for creating a remote system reset volume (a real drive is only necessary for reading files from a real diskette). The public volume manager personal computer uses a command processor (VVOL.COM) to perform the following functions:

- Create virtual volume data sets on the 4700 fixed disks
- Access the VVOL.SYSTEM or VVOL.PUBLIC data sets for reading, writing, or copying files
- Rename virtual volume data sets
- Erase virtual volume data sets from the 4700 fixed disks.

Loading the Virtual Volume Device Driver

A line in the CONFIG.SYS file on either the remote reset volume (if remote IPL is used) or the IPL diskette loads the virtual volume device driver during the boot procedure. The line is:

DEVICE=[d:][path]VVOLDD.SYS

DEVICE= is required to specify that this is a device driver.

[d:][path] give the optional drive letter **d:** and DOS directory specifier for the disk or volume where the VVOLDD.SYS file resides. The drive specifier is not needed if the file is on the IPL volume. The directory specifier is not needed if the file is in the root directory.

VVOLDD.SYS is the literal virtual volume device driver filename.

See the IBM personal computer DOS *Technical Reference* manual for more information on loading device drivers.

Using the Virtual Volume Device Driver

Once the virtual volume device driver is installed, you access virtual volumes exactly as if they were personal computer disk volumes. The number of personal computer disk or diskette drives your personal computer has, the number of block-type device drivers loaded, and the order in which they are loaded determines the drive specifiers for virtual volumes.

If VVOLDD.SYS is the first block-type device driver loaded, it will access the public volume using the next letter higher than that assigned to the last physical drive present. The private volume has the next letter higher than the public volume.

Virtual Volume Device Driver Status Messages

When the virtual volume device driver detects an error while attempting to service a request from DOS, the driver returns a status code to DOS indicating the nature of the error. If used, the DOS critical error handler will display a message. A user-written critical error handler will receive the error status in the low order byte of the DI register and can examine this value to determine what action to take. The error status refers to an error in the 4700 controller or in the communication hardware.

The DOS status message is in one of these forms:

<type> error reading <device>
or
<type> error writing <device>

<type> is the error type. DOS displays the error type based on the error code that the device driver returns.

<device> shows which device caused the error. It will be a disk drive identifier such as C:.

The information that follows shows the virtual volume error status codes and the corresponding <type> messages that DOS displays. Also included is information on the probable cause of the error and possible steps to take to correct it.

VVOLDD Status	Message <Type>
00H	Write protect

If the error is on the PUBLIC or SYSTEM volume:

Cause: The personal computer attempting to write is not the public volume manager.

Action: Abort the current function. Retry using your private volume.

Cause: The public volume manager does not have exclusive use of the volume.

Action: Abort the current function. Use the VVOL program to gain exclusive use of the volume, then retry the function.

Cause: The data set is write protected by the 4700 controller.

Action: Use controller access method facilities to unprotect the data set. (For more information on unprotecting data sets, see the *IBM 4700 Finance Communications System, Controller Programming Library*).

If the error is on the PRIVATE volume:

Cause: The data set is write protected by the 4700 controller.

Action: Use controller access method facilities to unprotect the data set. (For more information on unprotecting data sets, see the *IBM 4700 Finance Communications System, Controller Programming Library*).

VVOLDD Status	Message <Type>
01H	Bad unit

Cause: DOS passed an invalid subunit number to the virtual volume device driver.

Action: Contact the dealer from whom you purchased DOS.

VVOLDD Status	Message <Type>
02H	Not ready

If the error occurs on the PUBLIC or SYSTEM volumes:

Cause: The public volume manager has exclusive use of the volume. No other personal computer can access the volume until the public volume manager releases it.

Action: Retry the request until (or when) the public volume manager releases the volume.

If the error occurs on a PRIVATE volume:

Cause: You do not have access to a private volume.

Action:

- Abort the current function. Use the VVOL program to access a private volume, then retry the function.
- Correct the 4700 CPGEN. (Refer to the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System* to see how to correct the 4700 CPGEN.)

VVOLDD Status	Message <Type>
03H	Bad command

Cause: DOS passed an incorrect command to the virtual volume device driver.

Action: Contact the dealer from whom you purchased DOS.

VVOLDD Status	Message <Type>
04H	Data

Cause: A media defect on the controller disk.

Action: Use the appropriate controller PDP to find and resolve the problem.

Cause: The 4700 controller port having the problem is not configured for virtual volume support.

Action: Correct and reassemble the CPGEN (see the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System*).

Cause: There is no communications support code in the personal computer. Virtual volume access requires either the financial input adapter card or the CCC.COM program.

Action:

- If the financial input adapter card is present, replace it.
- If the financial input adapter card is absent, stop the function, load CCC.COM, then retry the function.

Cause: The 3278/79 Emulation Adapter card is missing or defective

Action:

- If a 3278/79 Emulation Adapter card is present, replace it.
- If no 3278/79 Emulation Adapter card is present, install one.

Cause: The COAX attachment to the controller is broken or disconnected.

Action: Check the attachment and correct the problem.

Cause: There is a hardware communications problem in the controller.

Action: Follow the appropriate controller PDP to resolve the problem.

Cause: The device cluster adapter (DCA) optional module is not present in the controller.

Action: Ensure that the controller load procedure does not exclude the required modules (see the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System* for details).

VVOLDD Status	Message <Type>
06H	Seek

Cause: The virtual volume data set does not exist.

Action: Use the VVOL program to create the data set.

Cause: Problems on the controller fixed disk.

Action: Follow the appropriate controller PDP to correct the problem.

VVOLDD Status	Message <Type>
07H	Non-DOS Disk

Cause: The virtual volume was formatted with more than 81 cylinders and an attempt was made to access it using DOS 2.0 or 2.1.

Action: Upgrade to DOS version 3.0 or later.

VVOLDD Status	Message <Type>
08H	Sector not found

Cause: A hardware error on the controller disk (not a media error).

Action: Use the appropriate controller PDP to find and resolve the problem.

VVOLDD Status	Message <Type>
0CH	Disk

Cause: The virtual volume optional module is not present in the controller.

Action:

- Correct and reassemble the CPGEN (see the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System*).
- Ensure that the controller load procedure does not exclude the required modules (see the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System*).

Cause: System software did not observe the communications protocols.

Action: Notify your IBM service representative for finance system products.

Using the VVOL Command

The operator of the public volume manager personal computer enters the VVOL command on the DOS command line as follows:

[d:]VVOL [DATA-SET-NAME][cdrive] [OPTIONS]

[d:]

The optional drive identifier for the disk/diskette drive on which VVOL.COM resides.

VVOL

The command-line evocation of the VVOL command processor.

DATA-SET-NAME

The name of the 4700 data set that this command affects. The choices are:

- VVOL.PUBLIC (if VVOL.PUBLIC is to be renamed)
- The name of any private virtual volume data set. All names of 4700 data sets for use with the personal computer must begin with "VVOL."; the user can define the remaining 12 characters of the name.

cdrive

The optional controller drive identifier for the controller fixed disk on which the data set resides or will reside. This is only necessary for private volumes. The default is drive A.

OPTIONS

One or two byte codes that control the effect of the command. A slash character ("/") must precede the option code. A colon and a parameter value can follow the option code if required by the option requested.

Note: If the VVOL command is given with a private volume data set name, with or without a controller drive letter (the default is A), and no OPTIONS, the named data set becomes the private volume of the issuing personal computer. Any previously assigned private volume is released.

Any user has access to this form of the command. It is not restricted to the public volume manager.

Option Codes and Parameter Values

Code	Meaning
/R	Releases the private volume assigned to the personal computer; allows that volume to be used by another personal computer. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters. **Note:** Any user can invoke this option (as well as the no option form of the command). Only the public volume manager can invoke the remaining options.
/C:cyls	Create a private volume. The data set name can be any valid 4700 data set name beginning with "VVOL." except VVOL.SYSTEM and VVOL.PUBLIC. cyls is a decimal number of cylinders the data set is to contain on the virtual volume. A cylinder is 128K bytes. The default is 4 cylinders (512K). This command fails if the data set name already exists on the drive.
/CP:cyls	Create the public volume. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters. The DATA-SET-NAME is always VVOL.PUBLIC. The controller configuration determines the drive where the data set resides. cyls is a decimal number of cylinders the data set is to contain on the virtual volume. A cylinder is 128K bytes. The default is 8 cylinders. This command fails if VVOL.PUBLIC already exists.

/CS:dkt Create the system reset volume. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters. The DATA-SET-NAME is always VVOL.SYSTEM, and the drive is determined by the controller configuration.

dkt specifies the real diskette drive that contains a diskette of data for the remote system reset volume. The size of the remote system reset volume (VVOL.SYSTEM) will be the same as the size of the source diskette. Any personal computer DOS diskette can be used; VVOL.SYSTEM can be 160K, 300K, 320K, or 360K. This command fails if VVOL.SYSTEM already exists.

This command automatically releases the new VVOL.SYSTEM data set when it completes. Any personal computer can then use the VVOL.SYSTEM data set for remote IPL.

/E Erase a private volume data set. This option requires the DATA-SET-NAME and the "cdrive" parameters unless the data set is on the A: drive. The data set name can be any valid 4700 fixed disk data set name beginning with "VVOL." except for VVOL.SYSTEM and VVOL.PUBLIC.

/EP Erase the public volume data set. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters. You will normally only use this command to erase VVOL.PUBLIC so that you can create a new VVOL.PUBLIC data set that has a different size.

/ES Erase the system volume data set. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters. Use this command to erase VVOL.SYSTEM; then you can create a new VVOL.SYSTEM data set that has a different size or different contents.

/N:NEW-NAME
Rename a private virtual volume data set. This option causes the data set DATA-SET-NAME to be renamed NEW-NAME. Specify **cdrive** if the data set DATA-SET-NAME is not on controller drive A. **NEW-NAME** can be any valid virtual volume data set name except for VVOL.SYSTEM. You cannot create the system reset virtual volume by renaming an existing data set.

/RP Release the public volume for general use. The public volume manager personal computer uses this command to release VVOL.PUBLIC to general use after modifying or copying files. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters.

/RS Release the system volume for general use. The public volume manager uses this command to release VVOL.SYSTEM to general use after modifying or copying files. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters.

/XP Exclusive use of the public volume. This option captures the VIRTUAL.PUBLIC data set for exclusive read/write/modify use by the public volume manager. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters.

/XS Exclusive use of the system volume. This option captures the VIRTUAL.SYSTEM data set for exclusive read/write/modify use by the public volume manager. This option does ***not*** require the DATA-SET-NAME and "cdrive" parameters.

Note: You should only use the /XP and /XS options when no other personal computers are operating from the controller. If this is not possible, notify users of the other personal computers. During the time that you have exclusive use of a virtual volume, any other personal computer attempting to use the volume will receive "not ready" status.

VVOL.COM Messages

Messages are produced by VVOL.COM when:

- Errors are detected in the command syntax.
- I/O errors occur.
- Situations requiring user intervention or response arise.

All messages are contained in a file named VVOL.MSG that must be available in the current directory on the default drive when VVOL.COM is executed. The Application Services diskette contains files named VVOLxxx.MSG where 'xxx' is a number identifying the language in which the message text appears. One of these files is selected during customization and is renamed VVOL.MSG for inclusion on the operational diskette.

All messages are written to the standard output device. This is normally the display screen unless redirected by the user when the VVOL command is issued.

Each message begins with a 4-digit message number and the name of the module issuing the message. This information serves to identify the source of the message in the event that the command is issued from a program or from a BAT file with ECHO OFF.

If the command terminates because of an error or a user option, the number of the last message displayed is used as a return code. Return codes can be interrogated by the BATCH subcommands IF and ERRORLEVEL, and by the DOS function call 4DH if VVOL.COM was invoked from a program using DOS function call 4BH.

In the event that the message file is unavailable or unreadable when VVOL.COM is invoked, the following message is displayed:

9999 VVOL.MSG DOS RC = nn

where nn is replaced by a 1 or 2 digit DOS error code that indicates the problem encountered. These error codes are documented in the DOS technical literature. The most likely one is code 2, meaning 'File not found'.

Should this problem occur, VVOL.COM will continue to run. However, if VVOL.COM attempts to display any messages, it can display only the message number. The list on the following pages can be consulted to determine the missing message text.

Message	Text
0001	VVOL.COM: Error reading from controller

Cause: A message was successfully written to the controller, but an error occurred when trying to read the response. A part might have failed in the communications hardware in your machine or in the controller or the connecting cable, or the controller might have been so busy that your machine timed out before the controller sent its response.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

You can try the operation as often as you like, by pressing the Enter key. If the problem persists, press the Esc key to cancel the request and then have the communications hardware serviced.

Message	Text
0002	VVOL.COM: Error writing to controller

Cause: An error has prevented writing a message to the controller. The problem can be in the communications hardware in your machine or in the controller or the connecting cable. Or it might be that the controller configuration was incorrectly specified, or the cable was disconnected - causing the controller to lose contact.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

If the link has worked before, or if the cable was disconnected and then reconnected, press the Enter key to retry the function. This resets the communication hardware and simulates a power-on to the controller.

If the link has never worked, you can press the Enter key to retry the operation, several times if necessary. If the problem persists, press the [Esc] key to cancel the request. Then check the configuration specifications for the controller and make sure that the port to which your machine is connected was defined as requiring DCA support. If not, correct the configuration and restart. If so, have the communications hardware serviced.

Message	Text
0003	VVOL.COM: Controller virtual volume support not available for this port.

Cause: The port to which this machine is attached was not defined in the controller configuration process as one requiring virtual volume support.

Action:

- Correct the controller configuration.
- Disconnect this machine from its present port and connect it to one that was defined for virtual volume support. Then reissue the command.

Message	Text
0004	VVOL.COM: System error 4 has occurred. Notify your IBM support representative.

Cause: A message was sent to the controller, but the controller responded with status indicating that the message was invalid.

Action: Ask your IBM support representative to obtain a program correction.

Note: If the debugging aid BTRACE.SYS is in use, dump the trace log to a diskette file using the DTRACE.COM module, and give the diskette containing the dump to your IBM support representative.

Message	Text
0005	VVOL.COM: Data set does not exist.

Cause: An attempt was made to:

- Access a private volume
- Gain exclusive access to either the public or the system volume
- Rename a volume.

But, the data set containing the volume was never created or has been erased or renamed.

Action: If necessary, have the public volume manager create the volume. Then reissue the command.

Message	Text
0006	VVOL.COM: The volume is in use.

Cause: An attempt was made to access a private volume that is currently being used by another station.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

You can either:

- Wait until the other user is finished with the volume, and then press the Enter key to retry the access function, or
- Press the Esc key to cancel the request so you can continue using your machine. Later, after the other user is finished with the volume, reissue the command.

Message	Text
0007	VVOL.COM: System error 7 has occurred.

Cause: Unexpected status (write protect violation) was received from the controller in response to a request. Under no circumstance is this a valid response to a request from this program.

Action: Ask your IBM support representative to obtain a program correction.

Note: If the debugging aid BTRACE.SYS is in use, dump the trace log to a diskette file using the DTRACE.COM module, and give the diskette containing the dump to your IBM support representative.

Message	Text
0008	VVOL.COM: A data error (CRC) has occurred on the controller data set

Cause: While attempting to read or write to a virtual volume data set, the controller detected a CRC error that could not be recovered.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

Press the Enter key to retry the function, several times if necessary. If the problem persists, press the Esc key to cancel the request. Then, refer to controller problem resolution procedures for handling disk CRC errors.

Message	Text
0009	VVOL.COM: A controller disk error has occurred.

Cause: While attempting to read or write to a virtual volume data set, the controller detected a disk error that could not be recovered.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

Press the Enter key to retry the function, several times if necessary. If the problem persists, press the Esc key to cancel the request. Then, refer to controller problem resolution procedures for handling disk errors.

Message	Text
0010	VVOL.COM: Data set already exists.

Cause: An attempt was made to create a virtual volume data set, and the target drive already contains a data set having that name; or an attempt was made to rename a data set, and a data set having the proposed new name already exists on the indicated drive.

Action: Reissue the command using a different name or a different drive; or erase or rename the existing data set having the desired name, and then reissue the command.

Message	Text
0011	VVOL.COM: Insufficient space for new data set on controller disk

Cause: An attempt was made to create a virtual volume data set; and the controller determined that the directory on the target disk was full, or there was insufficient space on the target disk to contain a data set of the given size.

Action: Erase one or more existing data sets and reissue the failing command, or reissue the failing command specifying a smaller size or a different drive.

Message	Text
0012	VVOL.COM: Request valid only for the public volume manager

Cause: A command reserved for the public volume manager was issued from a machine attached to a port other than the one designated for use by the public volume manager. The only functions available to users other than the public volume manager are the accessing and releasing of private volumes.

Action: Have the public volume manager issue the required command using the machine attached to the designated port.

Message	Text
0013	VVOL.COM: FIO adapter card required or module CCC.COM must be resident

Cause: Communication with the controller requires the presence of either the financial input option adapter or the module CCC.COM.

Action: If the financial input adapter card is present, replace it. If not, install a financial input adapter card or load module CCC.COM. Then reissue the command.

Message	Text
0014	VVOL.COM: VVOLDD.SYS device driver required

Cause: Execution of this command requires the presence of the VVOLDD.SYS device driver.

Action: If a file named CONFIG.SYS exists on your system volume, add a line to it as follows:

DEVICE=[d:][path]VVOLDD.SYS

If no CONFIG.SYS file exists, create one having the line shown above.

After creating or appending to CONFIG.SYS, use Alt-Ctrl-Del (or turn power off and on again) to reload DOS with the needed device driver. Then reissue the command.

Message	Text
0015	VVOL.COM: Unknown diskette media. Replace diskette in drive %s.

Cause: The system diskette in the indicated drive has an unrecognized format (the %s in the message is replaced by a drive letter). Either the sector size is not 512 bytes, or the media descriptor byte in the boot sector does not match any of those produced by the FORMAT command of DOS version 2.0 or later.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

If the intended diskette is not in the indicated drive, insert the correct diskette and press Enter.

If the intended diskette is in the indicated drive, make sure that it was formatted using DOS version 2.0 or later. If necessary, press Esc to cancel the request, and then re-create the system diskette using a freshly formatted DOS diskette. Then reissue the command.

Message	Text
0016	VVOL.COM: Diskette drive %s is not ready

Cause: DOS returned a 'not ready' indication on an attempt to access the diskette in the indicated drive (the %s in the message is replaced by a drive letter).

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

Make sure the diskette is fully inserted, is right side up, and correctly oriented. Also ensure that the drive door is closed. Then press the Enter key to retry the function. If the problem persists, have the diskette drive/adapter serviced.

If you want to cancel the function, press the Esc key.

Message	Text
0017	VVOL.COM: Diskette in drive %s is unreadable

Cause: DOS returned a 'CRC error' or 'sector not found' indication on an attempt to access the diskette in the indicated drive (the %s in the message is replaced by a drive letter). These are usually media failures, which are sometimes temporary.

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

Press the Enter key to retry the function, several times if necessary. If the problem persists, press the Esc key to cancel the function. Then, re-create the system diskette using a freshly formatted DOS diskette and reissue the command.

Message	Text
0018	VVOL.COM: Diskette hardware error on drive %s.

Cause: DOS returned an error code that usually signifies a hardware error in the diskette drive or the controller card (the %s in the message is replaced by a drive letter).

Action: This message is followed by message 0042, giving you the option to retry the function or to cancel the request.

Press the Enter key to retry the function, several times if necessary. If the problem persists, press the Esc key to cancel the function. Then, have the system serviced to correct the failure.

Message	Text
0019	VVOL.COM: Invalid diskette drive specified

Cause: Either a drive letter other than A, B, C or D was entered, or DOS does not recognize the the drive letter as a valid diskette drive for this machine.

Action: Reissue the command, either specifying a valid diskette drive or omitting the drive parameter and accepting the default value.

Message	Text
0020	VVOL.COM: No parameters were given

Cause: The command was entered with no virtual volume data set name and no option code. It has no meaning in this form.

Action: Check the syntax of the command, and reissue it with the required parameters.

Message	Text
0021	VVOL.COM: An option code was expected

Cause: A '/' character was found by itself, either at the end of the command line or followed by a delimiter. The correct syntax calls for the '/' to be followed immediately by the option code with no intervening delimiters.

Action: Reissue the command with the correct syntax.

Message	Text
0022	VVOL.COM: The option code is invalid

Cause: The character immediately following the '/' character on the command line was not one of C, R, X, E or N. These are the only valid options.

Action: Check the syntax of the command, and reissue it with the required parameters.

Message	Text
0023	VVOL.COM: A delimiter was expected

Cause: A character other than a delimiter was found at a point in the option field where the command was already logically complete.

Action: Check the syntax of the command, and reissue it with the required parameters.

Message	Text
0024	VVOL.COM: Too many parameters, the extra ones are ignored

Cause: A data set name was entered with a command that does not require one, or more than 3 parameters were entered, or a parameter was found after the option field.

Action: None. The command is performed as though the unnecessary parameters had not been entered.

Message	Text
0025	VVOL.COM: Controller drive letter invalid

Cause: A controller drive letter was entered with a data set name, and either the drive letter was not A, B, C, or D, or the controller configuration does not include the specified drive.

Action: Reissue the command, either specifying a valid drive or omitting the drive letter and accepting the default value.

Message	Text
0026	VVOL.COM: A '/' was expected

Cause: The third parameter, which must be the option field, did not begin with a '/' character as required.

Action: Check the syntax of the command and reissue it with the required parameters.

Message	Text
0027	VVOL.COM: A ':' was expected

Cause: The option field contained a /CP or a /CS or a /N followed by another character which was not a colon. The colon is the only valid character, other than a delimiter, allowed in this position.

Action: Check the syntax of the command and reissue it with the required parameters.

Message	Text
0028	VVOL.COM: Data set name exceeds 17 characters

Cause: The data set name parameter, or the new data set name parameter for the rename function, is longer than the maximum of 17 characters.

Action: Reissue the command with a valid data set name.

Message	Text
0029	VVOL.COM: Data set name must begin with 'VVOL.'

Cause: The first 5 characters of an entered data set name were not 'VVOL.' as required. The only controller data sets that can be manipulated with this command are those whose names begin with this character sequence.

Action: Reissue the command with a valid data set name.

Message	Text
0030	VVOL.COM: A 1-3 digit number was expected

Cause: The value for the number of cylinders parameter of the create function was longer than 3 characters.

Action: The valid range for this parameter is 1-255. Reissue the command using a 1-3 digit number for the cylinders parameter.

Message	Text
0031	VVOL.COM: Cylinders parameter must be in the range 1-255

Cause: The value for the number of cylinders parameter of the create function was 0 or greater than 255, outside the allowable range.

Action: Reissue the command specifying a valid number of cylinders.

Message	Text
0032	VVOL.COM: A decimal digit was expected

Cause: In the number of cylinders parameter of the create function, a character other than a decimal digit (0-9) was found.

Action: Reissue the command specifying the number of cylinders in decimal digits.

Message	Text
0033	VVOL.COM: A data set name is required

Cause: The command was interpreted as a request to create or erase a private volume data set, but the name of the data set was not entered.

Action: Reissue the command, specifying the name of the data set to be created or erased.

Message	Text
0034	VVOL.COM: The current data set name is required

Cause: The command was interpreted as a request to rename a data set, but the name of the existing data set was not entered.

Action: Reissue the command specifying the name of the data set to be renamed.

Message	Text
0035	VVOL.COM: A new data set name is required

Cause: The command was interpreted as a request to rename a data set, but the new name for the data set was not entered after the colon.

Action: Reissue the command specifying the new name for the data set.

Message	Text
0036	VVOL.COM: Cannot rename the Remote System Reset Volume

Cause: An attempt was made to rename the remote system reset volume. Since this is a special purpose data set in a format different from that of other virtual volumes (diskette, rather than disk) and is created using a special function, renaming it is not allowed.

Action: Use the DOS DISKCOPY program to make a backup copy of the system volume on a real diskette, then use the /ES option of the VVOL command to erase the virtual volume. You can then create a new system volume, or restore the old one. Use the VVOL command with the /CS option, and either a newly created system diskette or one previously backed up.

Message	Text
0037	VVOL.COM: Cannot create Remote System Reset Volume via RENAME

Cause: An attempt was made to rename a virtual volume data set on the same controller drive as the remote system reset volume, using VVOL.SYSTEM as the new data set name. The remote system reset volume can be created only from a real diskette using the /CS option of the VVOL command.

Action: If a new system volume is desired, use the /CS option of the VVOL command to create it from a physical diskette. Otherwise, reissue the command specifying a different data set name than VVOL.SYSTEM.

Message	Text
0038	VVOL.COM: A diskette drive letter was expected

Cause: The /CS option was entered followed by a colon and a delimiter. If the colon is used, it must be followed immediately by the letter designating the diskette drive that contains the new system diskette.

Action: If the default drive was intended to be used, reissue the command and omit the colon. Otherwise, reissue the command, and specify the drive letter immediately following the colon.

Message	Text
0039	VVOL.COM: A number of cylinders was expected

Cause: The /C or /CP option was entered followed by a colon and a delimiter. If the colon is used, it must be followed immediately by the number of cylinders desired in the data set.

Action: If the default number of cylinders was intended to be used, reissue the command, and omit the colon. Otherwise, reissue the command, and specify the desired number of cylinders immediately following the colon.

Message	Text
0040	VVOL.COM: A volume identifier ('P' or 'S') was expected

Cause: The form of the command implied that the public or system volume was being referred to, but the second character of the option field was neither 'P' nor 'S'. This character must be correctly entered to identify which volume to use.

Action: Check the syntax of the command and reissue it with the required parameters.

Message	Text
0041	VVOL.COM: Insert system diskette in drive %s

Cause: The program is ready to read the diskette and create a new system volume from its contents (%s in the message is replaced by a drive letter).

Action: None. This message is followed by message 0044, to which a reply is expected.

Message	Text
0042	VVOL.COM: Press [Enter] to RETRY, [Esc] to QUIT.

Cause: This message is preceded by a message indicating that an error has occurred and specifying the nature of the error. You have the option of retrying the operation that failed or of cancelling the command.

Action: See the description of the companion message for a discussion of the options.

If you want to cancel the command, press the Esc key. If you want to retry the failed operation, press the Enter key. If the error is a diskette error, you may want to replace the diskette before trying the operation again.

If you press any other key, the message repeats until one of the two expected keys is pressed.

Message	Text
0043	VVOL.COM: Request terminated by operator

Cause: You pressed the Esc key in response to a message that gave the option to continue or to cancel the command.

Action: None. The message is for information only.

Message	Text
0044	VVOL.COM: Press [Enter] when READY, [Esc] to QUIT.

Cause: This message follows message 0041. The program is ready to read the system diskette and is giving you the option of continuing with the command as entered or cancelling the command.

Action: To continue with the command, insert the system diskette into the indicated drive, and press the Enter key. To cancel the command, press the Esc key.

If you press any other key, the message repeats until one of the two expected keys is pressed.

Message	Text
0045	VVOL.COM: Please confirm request to ERASE data set %s.

Cause: The %s in the text shown is replaced by the name of a virtual volume data set that you have asked to erase. If the command is allowed to proceed, all data on the volume contained in the named data set will be lost.

Action: None. This message is followed by message 0046, to which a reply is expected.

Message	Text
0046	VVOL.COM: Press [Enter] to ERASE the data set, [Esc] to QUIT.

Cause: You have asked to erase a virtual volume data set, and the program is giving you a chance to change your mind before all data in the data set is lost.

Action: If you want to erase the data set, press the Enter key. If you want to keep the data set, press the Esc key.

If you press any other key, the message will be repeated until one of the two expected keys is pressed.

Examples Using the VVOL Command

This example demonstrates how to use the VVOL command to create a private volume with the following characteristics:

- The private volume is located on controller drive B.
- Its capacity is 512K (4 cylinders).
- Its name is VVOL.USERONE.

Enter at the DOS prompt:

VVOL VVOL.USERONE B /C:4

This command assumes that the VVOL command file is in the current default personal computer disk and directory.

The next example demonstrates how to use the VVOL command to create the system reset volume (VVOL.SYSTEM), using the diskette in personal computer drive A: as the source of the contents of VVOL.SYSTEM.

Enter at the DOS prompt:

VVOL /CS:A

This command assumes that the VVOL command file is in the current default personal computer disk and directory.

This example demonstrates how to use the VVOL command to get read/write/modify use of the public volume (VVOL.PUBLIC). Only the public volume manager can use this option.

Enter at the DOS prompt:

VVOL /XP

This command assumes that the VVOL command file is in the current default personal computer disk and directory.

After all modifications to the volume have been made, enter the command:

VVOL /RP

This releases the public volume for use by other personal computers.

The next example demonstrates how to use the VVOL command to erase a private volume:

- The private volume data set name is VVOL.USERONE.
- It resides on drive B.

Enter at the DOS prompt:

VVOL VVOL.USERONE B /E

This command assumes that the VVOL command file is in the current default personal computer disk and directory. Only the public volume manager can use this option.

Section 7. Diagnosis Guide

You can use this diagnosis section in two ways. Programmers and operations specialists can use the message section for reference, and to make preliminary determinations regarding the nature of a problem before notifying their Technical Coordinator. The Technical Coordinator, unable to solve a problem using available information, uses this section to develop a keyword string for reporting a problem to IBM.

When to Use Diagnosis Guide Tasks

When you experience a problem with Application Services, you will need to diagnose the problem. Problem diagnosis is the process of isolating the most likely source of a problem. The problem can be in a hardware unit, a software area, a communication facility, or a specific publication. The tasks in the diagnosis guide start with problems in the software area of Application Services.

The tasks that follow define the problem determination procedures. Detailed explanations of the procedures are on following pages.

- Problem Source Identification (PSI)

  In this task you determine whether the source of the problem is in Application Services or elsewhere. If the problem is not in the Application Services, consult other diagnostic aids. If the problem is in Application Services, the following steps apply:

- Build a keyword string.

  Here you follow steps to create a character string that will describe the problem to an IBM Support Center.

- Report the problem to an IBM Support Center.

- Work with the support center to submit an APAR (Authorized Program Analysis Report) if necessary.

Problem Source Identification

Problem Source Identification (PSI) determines whether the source of your problem is the 4700 Personal Computer Application Services.

Diagnosing Problems

It is possible that problems you discover while using Application Services are not caused by the personal computer. Thus, your first task in PSI is to decide whether or not the personal computer is the most likely source of the problem. You can do this by performing the tasks below:

1. Compare the **major** indication of your problem to each of the following:

   - Message: An unexpected or incorrect message is displayed on the personal computer's display station.

   - Incorrect output: A message, data, or return code printed on the personal computer's display is incorrect, incomplete, or incorrectly formatted.

   - Loop: A program is in a loop. The program is doing repetitive processing, or the system has stopped for no apparent reason.

   - Documentation error: Information within a publication is incomplete, inaccurate, or unclear.

2. If your symptom is described above, then the Application Services is the most likely source of your problem. See if you can resolve the problem; go to "Keyword String Selection" on page 7-5 to begin building your keyword string.

3. If one of the above does not describe your problem, then Application Services is probably not the source of your problem. Instead, one of the following could be the source:

   - A personal computer application program

     If a problem occurs while you are running a financial application program, check to see that it is working correctly.

   - Your PC DOS system.

     It may be necessary to use system facilities to determine the source of a system failure. Consult the *IBM PC DOS Technical Reference* for more information.

Keyword String Selection

During Problem Source Identification, you determined that the most likely source of your problem is the 4700 Personal Computer Application Services. Your next task is to describe the problem as a keyword string. A keyword string is a list of arguments that describes the program and its problem symptoms. This section tells you how to build a keyword string.

Keyword String Identification

A complete keyword string consists of the following arguments:

- Part Identification Number (required keyword)

  This is the Application Services software identifier.

  Note: The Part Identification Number for the 4700 Personal Computer Application Services is "6934406".

- Type of failure

  This is the major symptom of the problem.

- Function that failed.

  This is the function attempted by the user.

Symptom Selection

Begin building the keyword string with the symptom. Select from the list below the most appropriate description of the problem; then turn to the section indicated on one of the following pages.

- Symptom: Unexpected or incorrect message.

  Go to "MSG Procedure" on page 7-7.

- Symptom: Incorrect output displayed or returned.

  Go to "INCORROUT Procedure" on page 7-9.

- Symptom: Program is in a loop.

  Go to "LOOP Procedure" on page 7-11.

- Symptom: A document is incorrect or unclear.

  Go to "PUBS Procedure" on page 7-13.

MSG Procedure

An error in a message on the personal computer display can be:

1. An unexpected message. The message displayed is not what was expected for the action taken or the information entered.

2. An incorrectly formatted message. The message displayed is in an incorrect format; it could contain errors such as misspelled words or misprinted characters.

Note: Check to see that the incorrect message is not due to a user error. The format of the Application Services command that generated the error might be incorrect.

PROCEDURE:

1. Add the word "PART" to the Part Identification Number to form the first keyword. Report this as:

   PART=6934406.

2. Determine the function running by looking at the message from the procedure on the display.

   Note: Refer to Figure 7-1 on page 7-17 for a list of function keywords.

   Add the word "FUNCTION" to form the second keyword. Report this as:

   FUNCTION=function keyword.

3. Add the word "MSG" to the message number displayed on the display station or on the printed output to form the third keyword. Report it as:

   MSG=nnnn mod where nnnn is the message number and mod is the module

4. For messages in each category (Send, Receive, Virtual Volume, and High Level Interface), there is a number nnnn (0001,0002,...) and a function module name associated with the message text. For many of the messages, this number can represent a return code. For problem determination use, this return code has the same meaning as the corresponding message.

5. Your completed keyword should look like this:

   PART=6934406 FUNCTION=function name MSG=nnnn mod.

6. See the section "Reporting the Problem" on page 7-15 for instructions on what to do with this developed keyword string.

INCORROUT Procedure

Use this procedure if the personal computer generates an unexpected display or printed output. The display or printed output can appear to be incorrect, incomplete, or incorrectly formatted, and can be a message, data, or a return code.

PROCEDURE:

1. Determine whether the incorrect output is a case of:

   - User error: The operator entered correct data.
   - Documentation error: A specific publication contains incorrect instructions or descriptions.
   - Programming problems: A problem occurred in a specific module.

2. Follow the appropriate procedure after you have determined the cause.

   - **User Error Procedure**

     This error might apply to any command; look at the section of this book that describes the specific function to check the format of the command that was entered.

   - **Documentation Error Procedure**

     Go to the "PUBS Procedure" section.

- **Programming Problem Procedure**

  - Add the word "PART" to the Part Identification Number to form the first keyword. Report this as:

    PART=6934406.

  - Determine which function is running by looking at the message from the procedure.

    Note: Refer to Figure 7-1 on page 7-17 for a list of function keywords.

    Add the word "FUNCTION" to this function name to form the second keyword. Report this as:

    FUNCTION=function keyword.

  - Add the word "INCORROUT" to form the third keyword. Report this as:

    INCORROUT.

  - Your completed keyword should look like this:

    PART=6934406
    FUNCTION=function name
    INCORROUT.

3. See the section "Reporting the Problem" on page 7-15 for instructions on what to do with this developed keyword string.

LOOP Procedure

The symptom of a loop is iterative (repetitive) processing by a program: the program may never return control, or it may take a longer time than usual for specific tasks.

PROCEDURE:

1. Add the word "PART" to the Part Identification Number to form the first keyword. Report this as:

   PART=6934406.

2. Determine which function is running by looking at the messages and procedures on the personal computer display.[1]

   Note: Refer to Figure 7-1 on page 7-17 for a list of function keywords.

   Add the word "FUNCTION" to this function name to form the second keyword. Report this as:

   FUNCTION=function keyword.

3. Add the word "LOOP" to form the third keyword. Report this as:

   LOOP

---

[1] If the screen is blank or if recognizable information has scrolled off the screen and is not reappearing as part of the loop, you should reinitialize the personal computer from IPL. Then attempt to recreate the problem, taking note of the sequence of events and the functions executing. If you have a printer, you can print the contents of each screen by using the DOS "CTL PRTSC" function.

4. Your completed keyword string should look like this:

   PART=6934406
   FUNCTION=function keyword LOOP.

5. See the section "Reporting the Problem" on page 7-15 for instructions on what to do with this developed keyword string.

PUBS Procedure

A personal computer publication is incomplete, inaccurate, or unclear.

Note: Report documentation errors to the IBM Support Center only if the error affects the operation or use of the personal computer using Application Services. If the error does not affect the operation or use of Application Services, such as an unclear or inconsistent description of a function, the "Reader's Comment" form at the back of the publication should be used.

PROCEDURE:

1. Add the word "PART" to the Part Identification Number to form the first keyword. Report this as:

   PART=6934406.

2. Add the word "PUBS" to the publication number, omit the hyphens, and omit the version number to form the second keyword. (The version number is the last digit in the publication number.) Report this as an eight digit number:

   PUBS=publication number (as in PUBS=GC312090 for publication number GC31-2090-0).

3. Add the word "TYPE" to the type of publication to form the third keyword. Report this as:

   TYPE=SERVICE.

4. Determine the page in the publication where the problem occurs.

   Add the word "STATEMENT" and the page number to form the fourth keyword. Report this as:

   STATEMENT=page number in the publication.

5. Your completed keyword should look like this:

   PART=6934406 PUBS=publication number TYPE=type STATEMENT=page number in the publication.

6. See the section "Reporting the Problem" on page 7-15 for instructions on what to do with this developed keyword string.

Reporting the Problem

After you have built your keyword string, use it to report the programming problem to the IBM Support Center.

Software Support Facility

IBM maintains a Software Support Facility (SSF) that contains summary information for reported problems relating to all supported software. The IBM Support Center personnel use keyword strings to add new problems and possibly their solutions to the SSF data base. Any authorized person may search the SSF, but only the IBM Support Center can update it.

The action taken by the IBM Support Center when they receive your keyword string will depend on whether or not your keyword string sufficiently describes the problem and whether or not the problem was previously reported.

To report a problem with your IBM 4700 Personal Computer Application Services program, call *1-800-426-4622*. When you call, please refer to component ID 566902201 in lieu of this product's part number.

Previously reported problems are in the SSF data base, and a solution could be available. If your problem was not previously reported, the IBM Support Center might ask you to assist in submitting an Authorized Program Analysis Report (APAR) for your problem. In order to assist in the APAR, you must retain all failure-related information, such as the following:

- The operating environment
- The sequence of events leading up to the problem
- The different symptoms of the problem.

If appropriate, you must also retain:

- Any ABEND dump data
- Any trace information
- Any other information that you gathered in the process of diagnosing the problem.

Keep all of this information until your problem is resolved completely.

Keyword Table

Use this table to determine which function keyword to include in the keyword string.

Keyword	Function
FILE XFER	File Transfer
VVOL	Virtual Volume
HLLI	High Level Interface
SND	SEND
REC	RECEIVE

Figure 7-1. Chart of Functions and Function Keywords

The Trace Facility

The Trace facility enables you to generate a trace listing for programs and function calls in the personal computer. An IBM representative can request this information to help determine the cause of a problem.

Trace Facility Files

The trace facility consists of two files on the Application Services diskette. The files are:

1. BTRACE.SYS - allocates trace buffers and performs trace functions.

2. DTRACE.COM - dumps trace buffers to a file for later use.

Installing the Trace Facility

To install the trace facility, place the following statement in the CONFIG.SYS file:

DEVICE = [d:][path]BTRACE.SYS m n

[d:] is the disk/diskette identifier, and **[path]** is the directory specifier (both parameters are optional) for the BTRACE.SYS file.

m = the size of the function trace buffer in k-bytes.

n = the size of the loop trace buffer in k-bytes.

Install BTRACE.SYS first; this lets you trace the installation of the rest of the programs. If you do not code the "m" and "n" values, function tracing will default to a buffer of 2K bytes. If you do not code the "n" value, the default loop trace buffer size is 0 bytes, which disables loop tracing. If m plus n is greater than 56K bytes, the program sets n to 0 and m to 56K bytes or the specified value, whichever is less.

After installing the trace facility, dump the contents of the trace buffer to a file at any time by executing the following command on the DOS command line.

[d:][path]DTRACE [d:][path]filename[.ext]

d: is the optional drive identifier, and **[path]** is the optional directory specifier.

DTRACE is the trace buffer dump command.

filename[.ext] is the destination file for the trace buffer contents.

Section 8. Using the High Level Language Interface

The High Level Language Interface

The high level language interface programs (HLLI) for the personal computer allow your application programs to access 4700 Personal Computer Application Services function calls and DOS function calls without using assembler routines. The high level languages that HLLI supports are:

- COBOL
- BASIC
- Compiled BASIC
- Pascal
- C (Lattice[1] compiler, small model [2]).

HLLI provides a software interface to DOS function calls as described in the *DOS Technical Reference*, and Application Services and translation function calls as defined in other sections of this book. The HLLI code converts variables in the calling high level program into the form that a function call uses, and calls the function using the standard interrupt-register value method. If the function returns values, the HLLI code converts them into variables in the high level program. The high level program has complete access to the function calls.

---

1 Lattice is a registered trademark of Lattice, Inc.

2 64K data and 64K program space

HLLI consists of three parts:

- Language Part (LP)
- Resident Part (RP)
- Profile.

The Language Part (LP) consists of a small object module (about 800 bytes total). There is a different version of LP for each high level language supported. You link the proper version of LP with your application program before using HLLI (for BASICA you use BLOAD).

The Resident Part (RP) is a resident interrupt routine that you load before using any HLLI functions. RP remains resident in storage and is available to any application program in a supported language. RP uses INT 6BH and consists of about 4800 bytes of code, of which about 1700 bytes is executable code and the rest is the profile. RP formats application program function calls into a form that DOS or the Application Services functions can use.

The profile is a text file that defines how HLLI maps arguments between the high level language and the processor registers. This mapping is predefined for all standard supported function calls. You can define new HLLI functions using the profile. You can also take out unwanted functions. The profile supplied is user modifiable and is a general implementation of the interrupts. You should optimize the profile for your needs.

Loading the High Level Language Interface

Load the resident part (RP) of the high level language interface by invoking the IPTHLI.COM program, either at the DOS prompt or from the AUTOEXEC.BAT file at system initialization. The following line will load the high level language interface:

[d:][path]IPTHLI

[d:] is the optional identifier for the drive, and **[path]** is the optional directory specifier for the IPTHLI.COM file.

IPTHLI.COM automatically loads the HLLI profile from a file called IPTHLI.PRO. The IPTHLI.PRO program file must be on the default drive and directory when you run IPTHLI.COM. If there is an error in the IPTHLI.PRO file, the IPTHLI.COM program displays an error message and terminates without loading. Once the profile is loaded, it remains loaded until the next IPL. See "Defining the HLLI Profile" on page 8-17 for details on the HLLI profile.

Status and Error Messages

IPTHLI.COM displays these messages.

Message
0001 HLLI.COM - HLLI already loaded, request ignored

Cause: IPTHLI.COM) is installed.

Action: Use the loaded program.

Message
0002 HLLI.COM - HLLI NOT LOADED

Cause: An error in the profile file

Action: Correct the profile and reload HLLI.

Message
0003 HLLI.COM - Open error, file : IPTHLI.PRO

Cause: Error opening the IPTHLI.PRO file.

Action: See the error codes in message 0005.

Message
0004 HLLI.COM - Close error, file : IPTHLI.PRO.

Cause: Error closing the IPTHLI.PRO file.

Action: See the error codes in message 0005.

Message
0005 HLLI.COM - DOS returned an error code of nn

Cause: Error reading the IPTHLI.PRO file. nn is a DOS error code.

Action: Look in the DOS manual for the error code ss and take the instructions in the DOS manual.

Message
0006 HLLI.COM - HLLI profile error, record number xx

Cause: HLLI found an error in record number xx in the IPTHLI.PRO file. Error message 0008 will display the error record.

Action: Find and correct the error record in the profile. Reinstall HLLI.

Message
0007 HLLI.COM - Wrong length, record yy

Cause: Record number yy in IPTHLI.PRO is not 89 bytes long.

Action: Find and correct the error record in the profile. Reinstall HLLI.

Message
0008 HLLI.COM - <record>

Cause: The record displayed, <record>, contains an error (in IPTHLI.PRO).

Action: Same as message six.

Message
0009 HLLI.COM - HLLI loaded.....

Cause: HLLI has successfully loaded the profile file and is awaiting commands.

Action: None.

Arguments Passed by HLLI

The application program using HLLI must define a set of arguments for passing variables and data between the application and the function called. This section explains each argument in detail. See the high level language code examples ("High Level Language Code Examples" on page 8-29) for specific data declarations. See "Defining the HLLI Profile" on page 8-17 for details on the mapping of arguments between application programs and function calls.

The COMM Argument

The COMM argument consists of 44 bytes of data organized as 22 binary integers (22 words). See "HLLI Function Call Options" on page 8-45 for the arguments used by specific functions. Details on each COMM argument follow.

The RETURN Argument

The first word of the COMM field is the RETURN word. A function call returns information to the application program using this word. See the specific function call in "HLLI Function Call Options" on page 8-45 for possible return values and their meanings. If an error occurs in the HLLI program, the RETURN word will contain one of these values for all functions:

(-2) The STRING argument is not in the Name Table of the profile (see "Defining the HLLI Profile" on page 8-17).

(-3) HLLI cannot find the requested profile entry.

(-4) The profile error bit is on - no error register is specified.

(-5) IPTHLI.COM is not loaded.

The FCT Argument

The second word of the COMM area is the FCT (function code). Use it to send the two-byte integer value that requests a specific DOS or Application Services function.

The OPT Arguments

Use the nine option words OPT1 through OPT9 to send register values for a specific function to HLLI. The profile determines which OPT field loads to which register.

The OPT10 Argument

Set the tenth OPT word to 0001H to activate a function call trace. HLLI displays register values on the screen just before the function call interrupt and just after control returns from the function call.

Set the tenth OPT word to 0002H to activate the NO-OP interrupt. HLLI calculates all of the addresses as if the interrupt was being called but does not actually issue the interrupt. The specified return fields return the requested registers. You can use this to get address and segment registers for a user-defined function.

The ROPT1 Through ROPT10 Arguments

These fields contain the processor register values after the function call returns control to the calling program. The information in the ROPT fields is only valid if there was no error during the function call (indicated in the RETURN field).

The STRING Argument

This argument is not part of the COMM area. It is simply an ASCII string that is a separate argument to the HLLI call from the application. Use this field to select which processor interrupt to process for the function call. It can contain one of three literal strings, which must be in capital letters:

- "DOS" (selects INT 21H)
- "XLATE" (selects INT 6BH)
- "CAM" (selects INT 6FH).
- "2CAM" (selects INT 7AH).

Use the DOS interrupt for all standard DOS function calls. Use the XLATE interrupt for the Application Services translation function. Use the CAM interrupt for all 6FH Application Services functions. Use the 2CAM interrupt for all 7AH Application Services functions.

You can define another interrupt with its own string name by using the profile table. See "Defining the HLLI Profile" on page 8-17.

The ADR1 and ADR2 Arguments

Use these arguments to pass addresses to the function call. The two fields must always be specified, but can contain an address of zero (NULL) if no address is needed. ADR1 or ADR2 can specify the address of a buffer for input data from a function call, or a string or data array address to pass to a function call.

These arguments are not part of the COMM area, but are separate arguments to the HLLI function call.

Defining the HLLI Profile

The file IPTHLI.PRO contains the HLLI profile. This is an ASCII file that contains a set of specifications for mapping HLLI parameters to processor registers when calling a function and for mapping processor registers to HLLI parameters when returning from the function.

The profile defines the mappings for all of the supported functions (see "HLLI Function Call Options" on page 8-45). You can redefine these mappings. You can also create new functions and use the profile to map them to HLLI.

The profile consists of lines of text (fixed-length records) 89 characters long. A line with a semi-colon as the first character is a comment.

The profile is divided into sections, which include the:

- Name Table, which refers the STRING argument to a specific processor interrupt.
- DOS interrupt area, which defines the DOS function interface.
- Other interrupt areas, which define all other interrupts and can include:
  - Translation interrupts area, which defines the ASCII/EBCDIC translation function interface.
  - Application Services interrupts area, which defines the Application Services function interface.
  - User-added interrupts for new functions.

The Name Table

The profile IPTHLI.PRO contains a small table called the **name table**. The name table consists of a few lines that refer an interrupt number to the STRING parameter in the HLLI call from the high-level application program. HLLI attempts to match the STRING parameter with an entry in the name table. A successful match specifies the interrupt number for use in the function call and a subfunction or module identifier used by HLLI.

The name table consists of four fields separated by commas:

1. The name table identifier - the character "&".

2. The string name - 8 upper case characters (if the name is less than 8 characters, the remaining characters are blanks).

3. The interrupt number (upper case) - HLLI uses this number to search for the proper profile lines for the function call.

4. The subfunction number (upper case) - HLLI uses this number, along with the actual function number from the FCT parameter, to find the specific profile line for the function call.

The "&" at the beginning of a line in the profile always specifies that line as part of the name table. All name table entries must be grouped together at the end of the profile.

See the IPTHLI.PRO file on the Application Services diskette and the example below to see how the name table is organized.

```
; start of name table
;
; name (max of 8 chara)
; |
; |       interrupt number
; |       |
; |       |  module id
; |       |  |
&CAM     ,6F,00
&XLATE   ,6B,05
&2CAM    ,7A,00
```

Function Specifications

Each function call that the HLLI supports is defined by one line in the profile. A function profile line consists of 30 two-character fields; each field is separated by a comma or a dash. There are three basic field divisions. They are the:

1. Identification area
2. Input mapping area
3. Return mapping area.

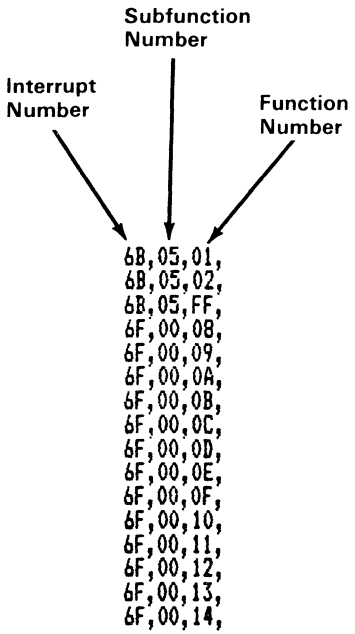
The Identification Area

The identification area consists of the first three two-character fields. This area identifies the function call that the rest of the line defines. The first field is the interrupt number, the second field is the subfunction number (00 if unneeded), and the third field is the function number. For example, the identification area for DOS function number 3FH (Read from file or device) would be:

```
21,00,3F,......
```

The above example specifies interrupt 21H, the standard DOS function call interrupt.

The IPTHLI.PRO file on the Application Services diskette and the example below show the organization of the identification area.



The Input Mapping Area

The input mapping area consists of the next 16 two-character fields. Each field specifies a processor register into which HLLI will place a specific parameter. The HLLI parameters defined in the input mapping area are, in order:

1. DSAP - the DS register
2. ESAP - the ES register
3. CSAP - the CS register

Note: HLLI sets up the above three parameters automatically using the values set by the application at the time of the HLLI call. The profile can map each of them into any valid register.

4. FCT - the function number
5. ADR1 - buffer address one
6. ADR2 - buffer address two
7. OPT1 - optional parameter 1

   :

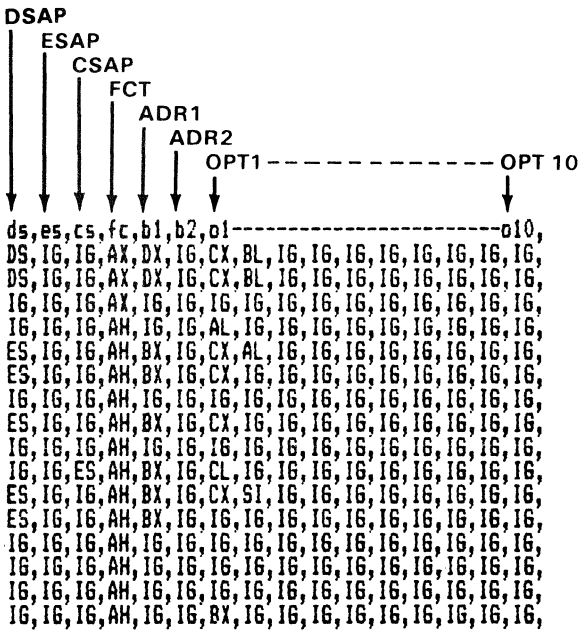
   :

16. OPT10 - optional parameter 10 (reserved for system use).

The processor registers that the profile can define for HLLI input parameters are the AX, AL, AH, BX, BL, BH, CX, CL, CH, DX, DL, DH, DI, SI, ES, and DS registers. The symbol "IG" specifies that the HLLI should ignore an input parameter. All register specifiers must be upper case.

When you use the profile with BASICA and compiled BASIC, a dash (instead of a comma) separating two fields defines the field preceding the dash as a string pointer. The dash should only appear after the ADR1 or ADR2 field (these are the only fields that should point to strings).

The IPTHLI.PRO file on the Application Services diskette and the example below show the organization of the input mapping area.



The Return Mapping Area

The return mapping area consists of eleven two-character fields at the end of the profile line. Each field specifies a processor register and the HLLI parameter that receives its value on return from the function. The fields of the return mapping area define (in order) processor registers:

1. AX
2. BX
3. CX
4. DX
5. DI
6. SI
7. ES
8. DS

The register fields accept ROPT1-ROPT10 as arguments. On function return, HLLI places the named register into the specified parameter. See the next page for specific field entries that specify ROPT1-ROPT10.

9. ER - This field accepts a processor register specification; HLLI places the specified register into the ERROR parameter.
10. EF - The processor flags register; the profile can contain a shift count value in this field - see the next page.
11. Spare - unused.

The list below shows the field entries that specify mapping of a register to a ROPT argument.

R1 to R10 Transfer all 16 bits to a ROPT parameter

L1 to L10 Transfer the lower 8 bits to a ROPT parameter

H1 to H10 Transfer the upper 8 bits to a ROPT parameter

S1 to S9 Transfer the high 8 bits to the RETURN option specified and transfer the low 8 bits to the next ROPT field. This option stores the 16-bit RETURN value in two consecutive ROPT fields.

IG Ignore.

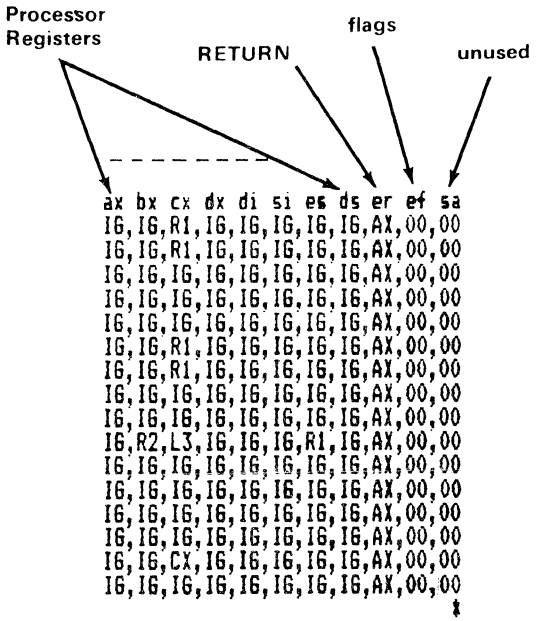
The EF profile field can contain a shift value. HLLI uses the shift value to determine which flag in the processor flags register to test. The mapping of the ER field to the ERROR parameter depends on the EF specification. The list below shows the possible shift values and the corresponding flag:

00H	No flag test
05H	OF flag
06H	DF flag
07H	IF flag
08H	TF flag
09H	SF flag
0AH	ZF flag
0CH	AF flag
0EH	PF flag
10H	CF flag.

The ER field accepts the same register definitions as those for the input mapping area of the profile. HLLI maps the specified register to the RETURN parameter depending on the profile entry for the EF field. If the EF field is "00", HLLI always returns the ER value if one is specified. If the EF field specifies a flag, the following combinations can occur on return from the function:

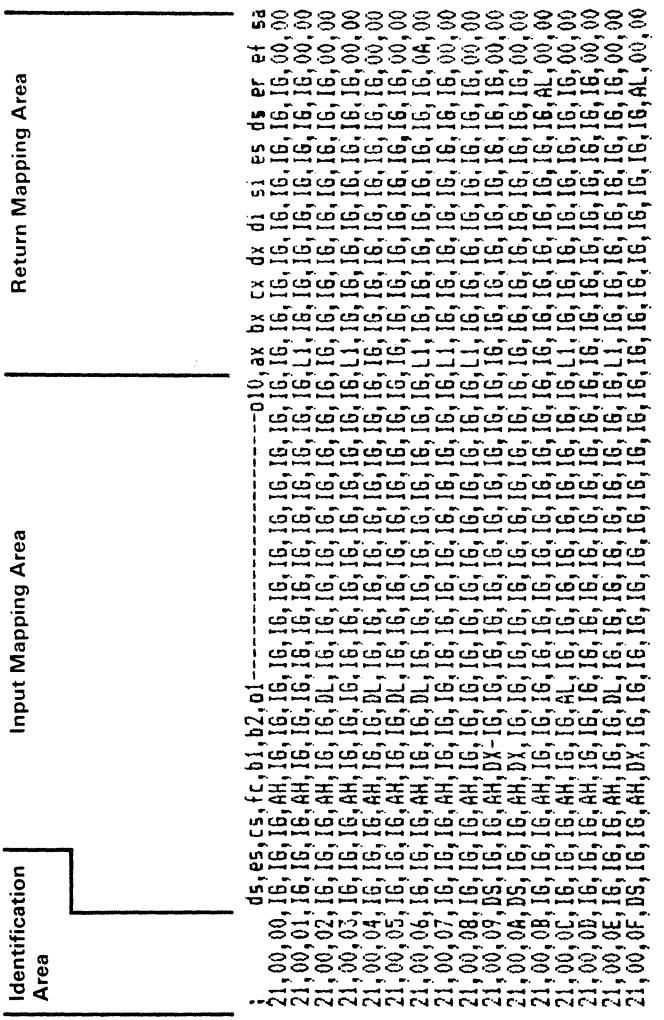
- **Flag set and ER register specified:** HLLI returns ER value in RETURN.
- **Flag set and no ER specification:** HLLI returns (-4) in RETURN.
- **Flag off and no ER specified:** HLLI returns 0000H in RETURN.

The IPTHLI.PRO file on the Application Services diskette and the example show the organization of the return mapping area.



Profile Example

The page below shows an actual section of the DOS function call profile part of IPTHLI.PRO. (This is subject to change).




```
;                ds,es,cs,fc,b1,b2,o1----------------------o10,ax bx cx dx di si es ds er ef sa
21,00,00,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,01,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,02,I6,I6,I6,AH,I6,I6,DL,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,03,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,04,I6,I6,I6,AH,I6,I6,DL,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,05,I6,I6,I6,AH,I6,I6,DL,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,06,I6,I6,I6,AH,I6,I6,DL,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,0A,00
21,00,07,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,08,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,09,DS,I6,I6,AH,DX,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,0A,DS,I6,I6,AH,DX,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,0B,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,AL,00,00
21,00,0C,I6,I6,I6,AH,I6,I6,AL,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,0D,I6,I6,I6,AH,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,0E,I6,I6,I6,AH,I6,I6,DL,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,00,00
21,00,0F,DS,I6,I6,AH,DX,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,I6,L1,I6,I6,I6,I6,I6,I6,I6,I6,AL,00,00
```

High Level Language Code Examples

These code examples show you how to use the high level language interface from each of the supported languages. For a general explanation of the argument-passing variables, which are similar for each language, see "Arguments Passed by HLLI" on page 8-13.

All arguments must be passed by their **address**.

All string arguments must terminate with the proper character for the function requested (this is usually a 0 byte).

The code in each of the language examples performs the same function. The code:

1. Opens a file called "HLI.COM" for reading and writing.
2. Reads 256 bytes from the file into a buffer.
3. Writes 256 bytes to the file from a buffer.
4. Closes the file.

COBOL

To use HLLI from a COBOL program, link IPTHLICO.OBJ with the COBOL program. Then you can call functions by using the following function call form:

```
CALL IPTHLI USING COMM,STRING,ADR1,ADR2
```

For real syntax, see the code example on the next page.

Guidelines:

- All fields of COMM must be COMP-0, as in the example.
- Byte precedence is reversed in COBOL. The other high level languages supported load a 2-byte integer into two consecutive bytes in storage low byte first; COBOL loads a 2-byte integer into storage high-byte first. HLLI compensates for this difference automatically, so you use identical values for function calls from any supported high level language.

COBOL Code Example

```
010 IDENTIFICATION DIVISION
020 PROGRAM-ID.        ME.
030 INSTALLATION.      HERE.
040     DATE-WRITTEN.     NOW.
050     SECURITY.          NONE.
060*
070*
080*
090 ENVIRONMENT DIVISION.
091     FILE CONTROL.
100 DATA DIVISION.
110     WORKING-STORAGE SECTION.
120 77 STING       PIC XXX VALUE 'DOS'.
130 01 OPENF       PIC 9 COMP-0 VALUE 61.
170 01 READF       PIC 9 COMP-0 VALUE 63.
205 01 WRITF       PIC 9 COMP-0 VALUE 64.
240 01 CLOSF       PIC 9 COMP-0 VALUE 62.
280 01 ATRIB       PIC 9 COMP-0 VALUE 02.
310 01 SIZRD       PIC 9 COMP-0 VALUE 0256.
360 01 FILEOT.
365* file to open followed by zero byte
370 03 FILEN       PIC X(9)  VALUE 'A:HLI.COM'
380 03 NIX         PIC 9 COMP-0 VALUE 0.
385* write buffer
390 01 BUFF1       PIC X(256) VALUE ZERO.
392* read buffer
395 01 BUFF2       PIC X(256) VALUE ZERO.
400 01 COMM.
405    03 RET      PIC 9 COMP-0.
410    03 FCT      PIC 9 COMP-0.
420    03 OPT1     PIC 9 COMP-0.
430       :
510    03 OPT10    PIC 9 COMP-0.
520    03 ROPT1    PIC 9 COMP-0.
530       :
610    03 ROPT10   PIC 9 COMP-0.
```

```
660 PROCEDURE DIVISION.
670 THIS-IS-IT.
700 MOVE OPENF TO FCT.      *OPEN function code
720 MOVE ATRIB TO OPT1.     *read/write file attributes
725* issue OPEN
730 CALL "IPTHLI" USING COMM, STING, FILEOT, buff2.
735 IF RET IS NOT EQUAL TO 0 GO TO ERROR.
740 MOVE ROPT1 TO OPT1.          *handle from DOS
750 MOVE READF TO FCT.           *READ function
760 MOVE SIZRD TO OPT2.          *read 256 bytes
765* issue READ
770 CALL "IPTHLI" USING COMM, STING, BUFF1, BUFF2.
775 IF RET IS NOT EQUAL TO 0 GO TO ERROR.
780 MOVE WRITF TO FCT.           *WRITE function
785* issue WRITE
790 CALL "IPTHLI" USING COMM, STING, BUFF1, BUFF2.
795 IF RET IS NOT EQUAL TO 0 GO TO ERROR.
800 MOVE CLOSF TO FCT.           *CLOSE function
805* issue CLOSE
810 CALL "IPTHLI" USING COMM, STING, FILEOT, BUFF2.
815 IF RET IS NOT EQUAL TO 0 GO TO ERROR.
820 STOP RUN.
830 ERROR.
840 DISPLAY "RETURN Error Code = ",RET
850 STOP RUN.
```

C

To use HLLI from a C program, link IPTHLIC.OBJ with your compiled C program. Call functions from the C program using the following form:

```
IPTHLIC(COMM,STRING,ADR1,ADR2);
```

For lines in real syntax, see the code example.

Note: The only version of C that HLLI supports is Lattice[3] C, small version (64K codespace and 64K data space).

---

3 Lattice is a registered trademark of Lattice, Inc.

C Code Example

```
main()
{
    static struct interface {   /* HLLI interface area */
         int    ret;      /* RETURN argument */
         int    fct;      /* function number */
         int    opt1;     /* OPT1 */
         int    opt2;
          :       :       /* thru */
         int    opt10;    /* OPT10 */
         int    ropt1;    /* ROPT1 */
         int    ropt2;
          :       :       /* thru */
         int    ropt10;   /* ROPT10 */
     } comm;
    static char buff1[257];  /* read buffer */
    static char buff2[257];  /* write buffer */

    /* set up and perform file open */

    comm.fct = 0x003d;
    comm.opt1 = 2;
    ipthlic(&comm,"DOS","a:hli.com",buff2);
    if (comm.ret != 0)
       printf("Error Return Code = %d",comm.ret);

    /* if no error, set up and perform read function */

    else {
         comm.fct = 0x03f;
         comm.opt1 = comm.ropt1;
         comm.opt2 = 0x0ff;
         ipthlic(&comm,"DOS",buff1,buff2);
         if (comm.ret != 0)
             printf("Error Return Code = %d",comm.ret);
```

```
    /* if no error set up, perform write function */

    else {
        comm.fct = 0x040;
        comm.opt2 = 0x0ff;
        ipthlic(&comm,"DOS",buff1,buff2);
        if (comm.ret != 0)
            printf("Error Return Code = %d",comm.ret);

        /* if no error set up and perform file close */

        else {
           comm.fct = 0x03e;/* CLOSE function */
           /* issue CLOSE */
           ipthlic(&comm,"DOS",buff1,buff2);
           if (comm.ret != 0)
             printf("Error Return Code = %d",comm.ret);
         }
      }
   }
}
```

BASICA

To use HLLI with an interpreted BASICA program, use the BLOAD statement to load IPTHLI.BAS as part of the program. Then call functions through HLLI using the following form:

```
Call (address of) HLLI(COMM,STRING,ADR1,ADR2)
```

See the code example for syntax.

Guidelines:

- The COMM area must be an array of integers.
- A data buffer must be an array.

BASICA Code Example

```
010 REM - BASIC HLLI Calls
020 CLEAR, &F800      ' any valid program space ok
030 DIM COMM%(22)     ' Communications parameters
040 DIM BUFF%(256)    ' buffer
050 DIM BUF2%(256)    ' second buffer
060 BUFSTR$=" "
070 ADR1$=" "
075 REM  Load BASIC lanuage part of HLLI
080 BLOAD "A:IPTHLIB.BAS",&HFC00
090 IPTHLI=&HFC00  ' any valid address ok,same as BLOAD
100 REM -  Define COMM Area arguments
110 RETN%=0:FCT%=1:OPT1%=2:OPT2%=3
120 OPT3%=4:OPT4%=5:OPT5%=6:OPT6%=7
130 OPT7%=8:OPT8%=9:OPT9%=10:OPT10%=11
140 ROPT1%=12:ROPT2%=13:ROPT3%=14
150 ROPT4%=15:ROPT5%=16:ROPT6%=17:ROPT7%=18
160 ROPT8%=19:ROPT9%=20:ROPT10%=21
170 REM
180 REM Clear COMM area to zeros
190 REM
200 FOR I%=1 TO 21
210 COMM%(i%)=0
220 NEXT I%
230 REM
240 REM set up call and open file
250 REM
260 STRFUN$="DOS"
270 COMM%(FCT%)=&H3D
280 COMM%(OPT1%)=&H02
290 ADR1$="A:HLI.COM" + CHR$(0)
400 CALL IPTHLI(COMM%(0),STRFUN$,ADR1$,BUF2%(0))
410 REM  Check for error
420 IF COMM%(RETN%) <> 0 THEN GOTO 920  ' check errors
```

```
450 REM  Set up and perform file read
460 REM
470 COMM%(RETN%)=0
480 COMM%(FCT%)=&H003F
490 COMM%(OPT1%)=COMM%(ROPT1%)
500 COMM%(OPT2%)=256
600 CALL IPTHLICB(COMM%(0),STRFUN$,BUFF%(0),BUF2%(0))
610 IF COMM%(RETN) <> 0 THEN GOTO 920 ' errors?
620 REM
630 REM  Set up and perform file write
640 REM
650 COMM%(RETN%)=0
660 COMM%(FCT%)=&H40
670 COMM%(OPT2%)=256
675 CALL IPTHLICB(COMM%(0),STRFUN$,BUFF%(0),BUF2%(0))
680 IF COMM%(RETN%) <> 0 THEN GOTO 920 ' errors?
690 REM
700 REM set up and perform file close
710 REM
720 COMM%(RETN%)=0
730 COMM%(FCT%)=&H3E
860 CALL IPTHLICB(COMM%(0),STRFUN$,ADR1$,BUF2%(0))
880 IF COMM%(RETN%) <> 0 THEN GOTO 920  'errors?
890
900 SYSTEM    ' exit
910
915 REM - Error escape
920 PRINT "RETN%=",COMM%(RETN%)
930 SYSTEM
```

Compiled BASIC

To use HLLI form a compiled BASIC program, link IPTHLICB.OBJ with the compiled BASIC program. Call functions using HLLI and the following form:

```
Call IPTHLICB(COMM,STRING,ADR1,ADR2)
```

See the code example for specific syntax.

Compiled BASIC Code Example

```
10      REM - COMPILED BASIC PROGRAM FOR DOS CALLS
            CLEAR, &HF800
            DIM COMM%(22)
            DIM BUFF%(256)
            DIM BUF2%(256)
            BUFSTR$=" "
            ADR1$=" "
            REM  Define COMM arguments
            RETN%=0:FCT%=1:OPT1%=2:OPT2%=3
            OPT3%=4:OPT4%=5:OPT5%=6:OPT6%=7
            OPT7%=8:OPT8%=9:OPT9%=10:OPT10%=11
            ROPT1%=12:ROPT2%=13:ROPT3%=14
            ROPT4%=15:ROPT5%=16:ROPT6%=17:ROPT7%=18
            ROPT8%=19:ROPT9%=20:ROPT10%=21

            REM  Clear COMM area

            FOR I%=1 TO 21
                    COMM%(i%)=0
            NEXT I%

            REM Set up and execute file open

            STRFUN$="DOS"
            COMM%(FCT%)=&H3D
            COMM%(OPT1%)=&H02
            ADR1$="A: HLI.COM"
            ADR1$=ADR1$+CHR$(00)
            CALL IPTHLICB(COMM%(0),STRFUN$,ADR1$,BUF2%(0)
            IF COMM%(RETN%) <> 0 THEN GOTO 20 ' Error?
```

```
            REM Set up and execute file read

            COMM%(RETN%)=0
            COMM%(FCT%)=&H3F
            COMM%(OPT1%)=HANDL%
            COMM%(OPT2%)=256
            CALL IPTHLICB(COMM%(0),STRFUN$,BUFF%(0),BUF2%(0))
            IF COMM%(RETN) <> 0 THEN GOTO 20 ' Error?

            REM Set up and execute file write

            COMM%(RETN%)=0
            COMM%(FCT%)=&H40
            COMM%(OPT2%)=256
            CALL IPTHLICB(COMM%(0),STRFUN$,BUFF%(0),BUF2%(0))
            IF COMM%(RETN%) <> 0 THEN GOTO 20 ' Error?

            REM Set up and execute file close

            COMM%(RETN%)=0
            COMM%(FCT%)=&H3E
            CALL IPTHLICB(COMM%(0),STRFUN$,ADR1$,BUF2%(0))
            IF COMM%(RETN%) <> 0 THEN GOTO 20 ' Error?

            SYSTEM ' Exit

            REM  Error escape

0       PRINT "RETN%=",COMM%(RETN%)
              SYSTEM
```

Pascal

To use HLLI from a Pascal program, link IPTHLIP.OBJ with your compiled Pascal program. Call functions through HLLI using the following form:

```
IPTHLIP(ADR COM, ADR STRING, ADR ADR1, ADR ADR2)
```

See the code example for syntax.

Guidelines:

- All strings must be fixed length (not dynamic).
- Note that the ADR operation, which gives the address of a variable, is provided by IBM PC Pascal Version 2.0. ADR is not a standard Pascal operation.

Pascal Code Example

```
program main;

(* declare argument types *)

TYPE
     comm_array = array[1..22] of integer;
     intx       = string(3);
     filex      = string(10);
     buff       = array[1..256] of char;

     c          = ADR of buff;
     a          = ADR of intx;
     b          = ADR of filex;
     d          = ADR of comm_array;

(* create arguments *)

VAR
     comm       : comm_array;
     int_id     : intx;
     file_id    : filex;
     buffer     : buff;

(* Declare the HLLI interface procedure *)

procedure ipthlip ( w:d; x:a; y:b; z:c) ; extern;
```

```
(* main program *)

begin

    (* set up and execute file open *)

    int_id := 'DOS';
    file_id := 'A:HLI.COM'*CHR(00);
    comm[2] := 61;
    comm[3] := 02;
    comm[12] := 1;
    ipthlip(ADR comm,ADR int_id,ADR file_id,ADR buffer );
    if comm[1] <> 0 then error(comm[1]);

    (* set up file read and read data *)

    comm[02] := 63;
    comm[03] := comm[13]
    comm[04] := 256;
    ipthlip(ADR comm,ADR int_id,ADR buffer,ADR buffer );
    if comm[1] <> 0 then error(comm[1]);

    (* set up file write and write data *)

    comm[2] := 64;
    comm[4] := 256;
    ipthlip(ADR comm,ADR int_id,ADR buffer,ADR buffer );
    if comm[1] <> 0 then error(comm[1]);

    (* set up file close and close file *)

    comm[2] := 62;
    ipthlip(ADR comm,ADR int_id,ADR file_id,ADR buffer);
    if comm[1] <> 0 then error(comm[1])
end.
```

HLLI Function Call Options

HLLI provides three types of function calls, each using a different interrupt:

- Standard DOS functions, using interrupt 21H
- 4700 Personal Computer Application Services functions, using interrupts 6FH or 7AH.
- ASCII/EBCDIC translation functions, using interrupt 6BH.

The information that follows gives a brief explanation of each of the HLLI arguments used by specific DOS, Application Services, and translation function calls.

This information applies only if you use the standard profile provided in the distribution diskette file IPTHLI.PRO. You can change the profile to alter what fields are used and to add new functions. If you do, the specification that follows will not be valid for the changed functions.

HLLI DOS Function Calls

This section lists each of the DOS function calls supported by HLLI along with the arguments passed and returned. The arguments are in HLLI format. See "Arguments Passed by HLLI" on page 8-13 for details on HLLI arguments. Consult the *IBM PC DOS Technical Reference* for details on the DOS function calls. The function call information that follows gives only the argument passing information necessary to use DOS function calls with HLLI.

Call each of these functions using "DOS" in the **STRING** parameter to HLLI. This selects INT 21H.

0000H Program Terminate

On Entry

FCT 0000H

On Return

RETURN None

0001H Keyboard Input

On Entry

FCT 0001H

On Return

RETURN 0000H if successful
Non-zero if error in request

ROPT1 00CCH - CC is input character from keyboard

0002H Display Output

On Entry

FCT	0002H
OPT1	00CCH - CC is character to display

On Return

RETURN	0000H if successful Non-zero if error in request

0003H Auxiliary Input

On Entry

FCT	0003H

On Return

RETURN	0000H if successful Non-zero if error in request
ROPT1	00CCH - CC is the character from the auxiliary device

0004H Auxiliary Output

On Entry

FCT	0004H
OPT1	00CCH - Character to display

On Return

RETURN	0000H if successful Non-zero if error in request

0005H Printer Output

On Entry

FCT	0005H
OPT1	00CCH - Character to print

On Return

RETURN	0000H if successful Non-zero if error in request

0006H Direct Console Input

On Entry

FCT	0006H
OPT1	00FFH

On Return

RETURN	0001H if character ready 0000H if character not ready
ROPT1	00CCH - CC is input character if ready

0006H Direct Console output

On Entry

FCT	0006H
OPT1	00CCH- CC is character to output to standard output device

On Return

RETURN	0000H if successful Non-zero if error in request

0007H Direct Console Input Without Echo

On Entry

FCT	0007H

On Return

RETURN	0000H if successful Non-zero if error in request
ROPT1	00CCH - character from standard input if ready

0008H Console Input Without Echo

On Entry

FCT	0008H

On Return

RETURN	0000H if successful Non-zero if error in request
ROPT1	00CCH - character from standard input if ready

0009H Print String

On Entry

FCT	0009H
ADR1	Address of character string terminated by "$"

On Return

RETURN	0000H if successful Non-zero if error in request

000AH Buffered Keyboard Input

On Entry

FCT	000AH
ADR1	Input buffer address (see DOS manual for format of input buffer)

On Return

RETURN	0000H if successful Non-zero if error in request

000BH Check Standard Input Status

On Entry

FCT	000BH

On Return

RETURN	00FFH if character is available from standard input 0000H if no character is available from standard input

000CH Clear Keyboard Buffer and Invoke a Keyboard Function

On Entry

FCT	000CH
OPT1	00CCH - function number to invoke

On Return

RETURN	0000H if request complete Non-zero if error in request
ROPT1	00CCH - CC is character from input if function 1, 6, 7, or 8

Use this function to change the current keyboard entry function or to clear the keyboard buffer before continuing buffered entry.

000DH Disk Reset

On Entry

FCT	000DH

On Return

RETURN	0000H if request complete Non-zero if error in request

000EH Select Drive

On Entry

FCT	000EH

On Return

RETURN	0000H if request complete Non-zero if error in request
ROPT1	00CCH - CC is total number of drives in system

000FH Open File

On Entry

FCT	000FH
ADR1	Address of FCB for unopened file (see the DOS reference for details on FCBs)

On Return

RETURN	0000H if file opened 00FFH if file not opened

0010H Close File

On Entry

FCT	0010H
ADR1	Address of FCB for open file (see the DOS reference for details)

On Return

RETURN	0000H if file correctly closed 00FFH if file not correctly closed (see the DOS reference for details)

0011H Search for First Entry

On Entry

FCT	0011H
ADR1	Address of FCB for unopened file (see the DOS reference for details)

On Return

RETURN	0000H if matching file found 00FFH if file not found

0012H Search for Next Entry

On Entry

FCT	0012H
ADR1	Address of FCB for unopened file (see the DOS reference for details)

On Return

RETURN	0000H if matching file found 00FFH if file not found

0013H Delete File

On Entry

FCT	0013H
ADR1	Address of FCB for unopened file

On Return

RETURN	0000H if file found and deleted 00FFH if file not found

0014H Sequential Read

On Entry

FCT	0014H
ADR1	Address of FCB for opened file

On Return

RETURN	0000H if completed successfully 0001H if EOF (no data read) 0002H if DTA is too small 0003H if EOF (partial record read, remainder zero bytes)

0015H Sequential Write

On Entry

FCT	0015H
ADR1	Name of opened FCB.

On Return

RETURN	0000H if completed successfully 0001H if diskette full (write canceled) 0002H if DTA is too small (write canceled)

0016H Create File

On Entry

FCT	0016H
ADR1	Address of opened FCB.

On Return

RETURN	0000H if completed successfully 00FFH if file not created

0017H Rename File

On Entry

FCT	0017H
ADR1	Address of modified FCB.

On Return

RETURN	0000H if completed successfully 00FFH if file not renamed

0019H Current Disk

On Entry

FCT	0019H

On Return

RETURN	0000H if completed successfully Non-zero if request error
ROPT1	00CCH - CC is current default drive

001AH Current Disk

On Entry

FCT	001AH

On Return

RETURN	0000H if completed successfully Non-zero if request error
ROPT1	00CCH - CC is current default drive

001BH Allocation Table Information

On Entry

FCT	001BH

On Return

RETURN	0000H if completed successfully Non-zero if request error
ROPT1	Size of a physical sector
ROPT2	00CCH - Number of sectors per allocation unit
ROPT3	Number of allocation units
ROPT4	Segment address of the FAT identification byte for the default drive
ROPT5	Offset address of the FAT identification byte for the default drive. ROPT4:ROPT5 is the address (segment:offset) of the FAT identification byte.

001CH Allocation Table Information for Specific Drive

On Entry

FCT	001CH
OPT1	00CCH - CC is the drive number (0=default, 1=A, etc.)

On Return

RETURN	0000H if completed successfully Non-zero if request error
ROPT1	Size of as physical sector
ROPT2	00CCH - Number of sectors per allocation unit
ROPT3	Number of allocation units
ROPT4	Segment address of the FAT identification byte for the default drive
ROPT5	Offset address of the FAT identification byte for the default drive. ROPT4:ROPT5 is the address (segment:offset) of the FAT identification byte.

0021H Random Read

On Entry

FCT	0021H
ADR1	Address of FCB for open file

On Return

RETURN	0000H if completed successfully 0001H if EOF (no data read) 0002H DTA too small 0003H EOF (partial record read and filled with zeros)

0022H Random Write

On Entry

FCT	0022H
ADR1	Address of FCB for open file

On Return

RETURN	0000H if completed successfully 0001H Diskette full 0002H DTA too small

0023H File Size

On Entry

FCT	0023H
ADR1	Address of FCB for unopened file

On Return

RETURN	0000H if directory entry found 00FFH if directory entry not found

0024H Set Relative Record Field

On Entry

FCT	0024H
ADR1	Address of FCB for opened file

On Return

RETURN	0000H if successful Non-zero if request error

0025H Set Interrupt Vector

On Entry

FCT	0025H
ADR1	Address offset of entry point for the interrupt vector. The address segment will be the processor CS register (segment:offset = CS:ADR1).

On Return

RETURN	0000H if successful Non-zero if request error

0027H Set Interrupt Vector

On Entry

FCT	0027H
ADR1	Address of the FCB for an open file
OPT1	Number of records to read

On Return

RETURN	0000H if successful 0001H EOF (no data read) 0002H DTA too small 0003H EOF (partial record read and filled with zeros)
ROPT1	Number of records read

0028H Random Block Write

On Entry

FCT	0028H
ADR1	Address of the FCB for an open file
OPT1	Number of records to write

On Return

RETURN	0000H if successful 0001H Diskette is full 0002H DTA too small

0029H Parse Filename

On Entry

FCT	0029H
ADR1	Address of the command line to parse
ADR2	Address of a memory buffer to receive an FCB
OPT1	00CCH Bit control value

On Return

RETURN	0000H No global filename characters found 0001H Global filename characters used in command line 00FFH Drive specifier invalid
ROPT1	Address segment to first character after parsed filename
ROPT2	Address offset to first character after parsed filename (segment:offset = ROPT1:ROPT2)
ROPT3	Address segment to first byte of formatted FCB
ROPT4	Address offset to first byte of formatted FCB (segment:offset = ROPT3:ROPT4)

002AH Get Date

On Entry

FCT	002AH

On Return

RETURN	0000H Successful Non-zero if error in request
ROPT1	00CCH CC is the day of the week (0 = Sun., 6 = Sat.)

ROPT2	BBCCH CC is the day of the month, BB is the month
ROPT3	Address segment to first byte of formatted FCB
ROPT4	Year

002BH Set Date

On Entry

FCT	002BH
OPT1	BBCCH CC is the day, BB is the month
OPT2	Year

On Return

RETURN	0000H Successful Non-zero if error in request
ROPT1	0000H Date was valid 00FFH Date was invalid

002CH Get Time

On Entry

FCT	002CH

On Return

RETURN	0000H Successful Non-zero if error in request
ROPT1	XXMMH MM is minutes (0-59), XX is hours (0-23)
ROPT2	SSZZH ZZ is hundredths of a second, SS is seconds (0-59)

002DH Set Time

On Entry

FCT	002DH
OPT1	BBCCH CC is minutes (0-59), BB is hour (0-23)
OPT2	DDAAH AA is hundredths of a second, DD is seconds (0-59)

On Return

RETURN	0000H Successful 00FFH Time was invalid

002EH Set / Reset Verify Switch

On Entry

FCT	002EH
OPT1	0000H Set verify off 0001H Set verify on

On Return

RETURN	0000H Successful Non-zero if error

002FH Set / Reset Verify Switch

On Entry

FCT	002FH

On Return

RETURN	0000H Successful Non-zero if error
ROPT1	Address segment of DTA
ROPT2	Address offset of DTA

0030H Get DOS Version Number

On Entry

FCT	0030H

On Return

RETURN	0000H Successful Non-zero if error
ROPT1	BBCCH CC is major version number, BB is minor version number
ROPT2	0000H
ROPT3	0000H

0031H Terminate Process and Remain Resident

On Entry

FCT	0031H
OPT1	00CCH CC is the exit code
OPT2	Memory size in paragraphs

On Return

RETURN	None

0033H CTL-Break Check

On Entry

FCT	0033H	
OPT1		
	0000H	Request for the current state
	0001H	Request to set the current state
OPT2		
	0000H	Set the current state off if OPT1 = 0001H
	0100H	Set the current state on if OPT1 = 0001H

On Return

RETURN		
	0000H	Current state is OFF, OPT1 was 0000H
	0001H	Current state is ON, OPT1 was 0001H

0035H Get Vector

On Entry

FCT	0035H	
OPT1	00CCH	CC is the interrupt number

On Return

RETURN	0000H	Request completed
		Non-zero if request error
ROPT1		Address segment of interrupt vector
ROPT2		Address offset of interrupt vector (interrupt vector address segment:offset is ROPT1:ROPT2)

0036H Get Disk Free Space

On Entry

FCT	0036H
OPT1	00CCH CC is the drive number (0 = default, 1 = A)

On Return

RETURN	FFFFH Drive invalid
ROPT1	Number of sectors / cluster
ROPT2	Available clusters
ROPT3	Clusters / drive
ROPT4	Bytes / sector

0038H Get Country Dependent Information

On Entry

FCT	0038H
ADR1	Address of buffer to store returned information
OPT1	00CCH CC is the country code
OPT2	Two-byte country code if OPT1 is 00FFH

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Country code

0138H Set Current Country

On Entry

FCT	0138H
OPT1	00CCH CC is the country code
OPT2	Two-byte country code if OPT1 is 00FFH
OPT3	FFFFH

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0039H Create Subdirectory (MKDIR)

On Entry

FCT	0039H
ADR1	Address of ASCII string terminated by 00H

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

003AH Remove Subdirectory (RMDIR)

On Entry

FCT	003AH
ADR1	Address of ASCII string terminated by 00H hex

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

003BH Change the Current Directory (CHDIR)

On Entry

FCT	003BH
ADR1	Address of ASCII string terminated by 00H hex

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

003CH Create a File

On Entry

FCT	003CH
ADR1	Address of ASCII string terminated by 00H hex
OPT1	Attribute of File

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	File handle

003DH Open a File

On Entry

FCT	003DH
ADR1	Address of ASCII string terminated by 00H hex
OPT1	CCOOH CC is the open mode (see DOS reference)

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	File handle

003EH Close a File Handle

On Entry

FCT	003EH
OPT1	File handle returned by OPEN (003DH) or CREATE (003CH)

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

003FH Read from a File or Device

On Entry

FCT	003FH
ADR1	Address of read data buffer
OPT1	File handle returned by OPEN or CREATE
OPT2	Number of bytes to read

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Number of bytes read

0040H Write to a File or Device

On Entry

FCT	0040H
ADR1	Address of buffer containing write data
OPT1	File handle returned by OPEN or CREATE
OPT2	Number of bytes to write

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0041H Delete a File from a Specified Directory (UNLINK)

On Entry

FCT	0041H
ADR1	Address of ASCII string terminated by a byte of 00H

On Return

RETURN	0000H Successful completion Non-zero if request error
ROPT1	Successful completion CCCCH Error code as defined in DOS reference

0042H Move File Read / Write Pointer (LSEEK)

On Entry

FCT	0042H
OPT1	File handle returned by OPEN or CREATE
OPT2	0C00 C is method of moving (0, 1, or 2)
OPT3	Distance to move in bytes (most significant word)
OPT4	Distance to move in bytes (least significant word). Total offset is OPT3:OPT4
ADR1	Address of ASCII string terminated by a byte of 00H

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	New data pointer if no error occurred (most significant word)
ROPT2	New data pointer if no error occurred (least significant word). Total pointer is contained in ROPT1:ROPT2.

0043H Change File Mode (CHMOD)

On Entry

FCT	0043H
ADR1	Address of ASCII string terminated by byte of 00H hex.
OPT1	0000H Return file attributes in ROPT2 0001H Set file attributes as described in OPT2
OPT2	CCCCH File attributes

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	File attributes if OPT1 set to 0001H

0044H I/O Control for Devices (Get Device Information)

On Entry

FCT	0044H
OPT1	0000H File handle returned by OPEN or CREATE

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	00CCH - Device information

0144H I/O Control for Devices (Set Device Information)

On Entry

FCT	0144H
OPT1	00CCH Drive number (0 = default, 1 = A, etc.)
OPT2	00CCH New device information

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	00CCH - Device information

0244H I/O Control for Devices (Read Control Information)

On Entry

FCT	0244H
ADR1	Address of buffer to receive data
OPT1	00CCH File handle returned by OPEN or CREATE
OPT2	00CCH Number of bytes to read

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Bytes transferred

0344H I/O Control for Devices (Write to Control Channel)

On Entry

FCT	0344H
ADR1	Address of data to write
OPT1	00CCH File handle returned by OPEN or CREATE
OPT2	00CCH Number of bytes to write

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Bytes transferred

0444H I/O Control for Devices (Read to Control Channel)

On Entry

FCT	0444H
ADR1	Address of buffer to receive read data
OPT1	00CCH Drive number (0 = default, 1 = A, etc.)
OPT2	00CCH Number of bytes to read

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Bytes transferred

0544H I/O Control for Devices (Write to Control Channel)

On Entry

FCT	0544H
ADR1	Address of data to write
OPT1	00CCH Drive number (0 = default, 1 = A, etc.)
OPT2	00CCH Number of bytes to write

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Bytes transferred

0644H I/O Control for Devices (Input Status)

On Entry

FCT	0644H
OPT1	File handle returned by OPEN or CREATE

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Device status - see DOS manual

0744H I/O Control for Devices (Output Status)

On Entry

FCT	0744H
OPT1	File handle returned by OPEN or CREATE

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Device status - see DOS manual

0844H I/O Control for Devices (Is DOS 3.0 block device changeable)

On Entry

FCT	0844H
OPT1	00CCH Drive number (0 = default, 1 = A, etc.)

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Device status: 0001H = fixed; 0 = removable

0B44H I/O Control for Devices (Change DOS 3.0 Sharing Retry Count)

On Entry

FCT	0B44H
OPT1	00CCH Drive number (0 = default, 1 = A, etc.)
OPT2	Number of loops
OPT3	Number of retries

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0045H Duplicate a File Handle (DOS 3.0 and Later)

On Entry

FCT	0045H
OPT1	File handle returned by OPEN or CREATE

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	New handle

0046H Force Duplicate a File Handle (CDUP)

On Entry

FCT	0046H
OPT1	File handle returned by OPEN or CREATE
OPT2	New file handle

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0047H Get Current Directory

On Entry

FCT	0047H
ADR1	Address of a 64-byte area of user memory. If the function is successful this buffer will receive an ASCII string that is the directory path to the current directory, terminated by a zero byte.
OPT1	00CCH Drive number (0 = default, 1 = A, etc.)

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0048H Allocate Memory

On Entry

FCT	0048H
OPT1	Number of paragraphs of memory to allocate

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Address segment of memory allocated
ROPT2	Size of largest block of memory available if allocation fails

0049H Free Allocated Memory

On Entry

FCT	0049H
OPT1	Address of memory to free

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

004AH Modify Allocated Memory Blocks (SETBLOCK)

On Entry

FCT	004AH
OPT1	Address of memory to free
OPT2	New requested block size in paragraphs

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Size of largest block of memory, if allocation fails

004BH Load or Execute a Program (EXEC)

On Entry

FCT	004BH
ADR1	Address of ASCII program name string terminated by 00H
ADR2	Address of parameter block
OPT1	00CCH function value (see DOS reference)

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Size of largest block of memory, if allocation fails

004CH Terminate and Exit (EXIT)

On Entry

FCT	004CH
OPT1	00CCH Return code (see DOS reference)

On Return

RETURN	None

004DH Get Return Code of a Sub-Process (WAIT)

On Entry

FCT	004DH

On Return

RETURN	BBCCH - CC is information from the terminating subprocess; BB is the termination code.

004EH Find First Matching File (FIND FIRST)

On Entry

FCT	004EH
ADR1	Address of ASCII program name and search string terminated by 00H.
OPT1	File search attribute

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	Field to put in OPT1 for function 4FH
ROPT2	Field to put in OPT2 for function 4FH

004FH Find Next Matching File

On Entry

FCT	004FH
OPT1	Field returned from function 4EH ROPT1
OPT2	Field returned from function 4EH ROPT2

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0054H Get Verify Setting

On Entry

FCT	0054H

On Return

RETURN	0000H Verify is OFF 0001H Verify is ON

0056H Rename a File

On Entry

FCT	0056H
ADR1	Address of ASCII string ended by 00H. Drive, path and filename to be renamed.
ADR2	Address of ASCII string ended by 00H. New drive, path, and filename.

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0057H Get/Set a File's Date and Time

On Entry

FCT	0057H
OPT1	File handle returned by OPEN or CREATE
OPT2	0000H Get date and time 0001H Set date and time from OPT1 and OPT2
OPT3	Time to be set if OPT2 = 0001H
OPT4	Date to be set if OPT3 = 0001H

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred
ROPT1	If OPT2 = 0100, the date from the internal table for the file
ROPT2	If OPT2 = 0100, the time from the internal table for the file

0059H Get Extended Error (DOS 3.0 and Later)

On Entry

FCT	0059H
OPT1	0000H

On Return

RETURN	0000H Successful completion Non-zero if request error
ROPT1	Extended error code
ROPT2	BBCCH CC is the suggested action, BB is the error class
ROPT3	00CCH CC is the locus

005AH Create a Temporary File (DOS 3.0 and Later)

On Entry

FCT	005AH
ADR1	Address of ASCII file name and directory string ended by 00H.
OPT1	File attribute

On Return

RETURN	0000H Successful completion CCCCH DOS error code if request error
ROPT1	File handle if function successful

005BH Create a New File (DOS 3.0 and Later)

On Entry

FCT	005BH
ADR1	Address of ASCII file name and directory string ended by 00H.
OPT1	File attribute

On Return

RETURN	0000H Successful completion CCCCH DOS error code if request error
ROPT1	File handle if function successful

005CH Lock/Unlock File (DOS 3.0 and Later)

On Entry

FCT	005CH
OPT1	File handle returned by OPEN or CREATE
OPT2	0000H Lock file 0001H Unlock file
OPT3	Offset high
OPT4	Offset low
OPT5	Length high
OPT6	Length low

On Return

RETURN	0000H Successful completion CCCCH DOS error code if error occurred

0062H Get Program Segment Prefix Address (PSP) (DOS 3.0 and Later)

On Entry

FCT	0062H

On Return

RETURN	0000H Successful completion Non-zero if error in request
ROPT1	The internal PSP segment address for the currently running process

XLATE Function Calls for HLLI

ASCII/EBCDIC translation function calls are part of the 4700 Personal Computer Application Services. Call them with INT 6BH; use the STRING argument "XLATE".

0001H ASCII to EBCDIC Translation Request

On Entry

FCT	0001H
ADR1	Address of data buffer to be translated
OPT1	Number of bytes to translate
OPT2	00CCH CC is translation table to use, from 00 to 07.

On Return

RETURN	CCDDH CC is completion code, DD is 00H if request serviced
ROPT1	Count of characters untranslated after a data check

0002H EBCDIC to ASCII Translation Request

On Entry

FCT	0002H
ADR1	Address of data buffer to be translated
OPT1	Number of bytes to translate
OPT2	00CCH CC is translation table to use, from 00 to 07.

On Return

RETURN	CCDDH CC is completion code, DD is 00H if request serviced
ROPT1	Count of characters untranslated after a data check

00FFH Function Active Query

On Entry

FCT	00FFH

On Return

RETURN	CCDDH CCH is 0H if function is ready, FFH if function not loaded

Application Services 6FH Function Calls

Call these Application Services functions using INT 6FH; "CAM" as the STRING argument.

0008H Present Keystroke

On Entry

FCT	0008H
OPT1	00CCH CC is the ASCII code to be sent

On Return

RETURN	0000H Successful completion 00FFH Illegal code specified

0009H Open Data Transfer Path

On Entry

FCT	0009H
ADR1	Address of work buffer
OPT1	Size of work buffer
OPT2	Data transfer option flags

On Return

RETURN	0000H Successful completion 00CCH Error code

000AH Read Data Block

On Entry

FCT	000AH
ADR1	Address of work buffer
OPT1	Size of work buffer

On Return

RETURN	0000H Successful completion 00CCH Error code
ROPT1	Size of receive buffer

000BH Test for Data Block

On Entry

FCT	000BH

On Return

RETURN	0000H Data waiting 0001H No data waiting 0002H Data block is message
ROPT1	Size of received data block in bytes or status if RETURN = 2

000CH Write Data Block

On Entry

FCT	000CH
ADR1	Address of write data buffer
OPT1	Size of data buffer

On Return

RETURN	0000H Successful completion 00CCH Error code

000DH Close Data Transfer Path

On Entry

FCT	000DH

On Return

RETURN	0000H Successful completion Non-zero if request error

000EH Establish Event Interrupt Handler

On Entry

FCT	000EH
ADR1	Address of interrupt routine (CS of caller used as segment address)
OPT1	0001H Data block received 0002H Screen image modified

On Return

RETURN	0000H Successful completion Non-zero if request error
ROPT1	Address segment of previous interrupt routine
ROPT2	Address offset of previous interrupt routine (address segment:offset of previous routine is ROPT1:ROPT2)
ROPT3	00CCH CC is previous event flag(s)

000FH Read Emulation Screen Data

On Entry

FCT	000FH
ADR1	Address of buffer to receive data
OPT1	Number of bytes to retrieve
OPT2	Virtual screen offset of first byte (0-1919)

On Return

RETURN	00CCH CC is a completion code

0010H Read Terminal Status Line

On Entry

FCT	0010H
ADR1	Address of 80-byte buffer to receive data

On Return

RETURN	0000H if successful Non-zero if request error

0011H Read Terminal Status

On Entry

FCT	0011H

On Return

RETURN	0001H Keyboard locked 0002H Controller not available

0012H Issue Power On Reset

On Entry

FCT	0012H

On Return

RETURN	0000H Successful completion Non-zero if request error

0013H Query Emulated Cursor Position

On Entry

FCT	0013H

On Return

RETURN	0000H Successful completion Non-zero if request error
ROPT1	BBCCH CC = cursor column (0-79) BB = cursor row (0-24)

0014H Issue Keyboard Scan Code

On Entry

FCT	0014H
OPT1	BBCCH CC = scan code, BB = shift state flag

On Return

RETURN	0000H Successful completion Non-zero if request error

0015H Obtain Application Services Version Number

On Entry

FCT	0015H

On Return

RETURN	0000H Successful completion
ROPT1	BBCCH - BB = major version number; CC = minor version number

0016H Read Emulation Screen with Attributes

On Entry

FCT	0016H
ADR1	Address of buffer to receive data

On Return

RETURN	0000H Successful completion

00FFH Query Presence

On Entry

FCT	00FFH

On Return

RETURN	0000H if CCC is present 00FFH if CCC is not present

Application Services 7AH Function Calls

Call these Application Services functions using INT 7AH; "2CAM" as the STRING argument.

Resolve Name

On Entry

FCT	0081H
ADR1	Address of parameter list

On Return

RETURN	1200H Successful completion
	122EH Invalid Gate name
ROPT1	0700H
ROPT2	Gate ID

Query Session ID

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0001H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for SESSMGR

On Return

RETURN

1200H Successful completion
1205H Invalid OPT4 specified
1207H Invalid OPT2 specified
1208H Invalid OPT2 specified

Query Session Parameters

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0002H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for SESSMGR

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified

Query Session Cursor

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	000BH
OPT2	8020H
OPT3	00FFH
OPT4	Resolved value for SESSMGR

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified

Connect to Keyboard

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0001H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for KEYBOARD

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Disconnect from Keyboard

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0002H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for KEYBOARD

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Write Keystroke

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0004H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for KEYBOARD

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Disable Input

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0005H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for KEYBOARD

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Enable Input

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0006H
OPT2	8020H
OPT3	0000H
OPT4	Resolved value for KEYBOARD

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Copy String

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0001H
OPT2	8020H
OPT3	00FFH
OPT4	Resolved value for COPY

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Read Operator Information Area

On Entry

FCT	0009H
ADR1	Address of parameter list
OPT1	0002H
OPT2	8020H
OPT3	00FFH
OPT4	Resolved value for OIAM

On Return

RETURN

1200H	Successful completion
1205H	Invalid OPT4 specified
1207H	Invalid OPT2 specified
1208H	Invalid OPT2 specified
1234H	Invalid OPT1 specified

Appendix A. Controller File Transfer Program

This appendix contains information for 4700 controller programmers who install the 4700 Personal Computer Application Services in a 4700 controller.

Installing the Controller File Transfer Programs

On the 4700 File Transfer diskette, you will find two application programs named IPTXFER and IPTXIT in the SYSAP data set. These applications are required on your 4700 controller operating diskette if you want to use the file transfer facilities (SEND and RECEIVE commands) provided on the 5-1/4 inch Application Services diskette.

Entries for IPTXFER and IPTXIT must be added to the SYSAP data set on your 4700 operating diskette. To do this use the Transfer AP function provided on the 4700 Installation Diskette. For a complete description of how to use the 4700 Installation Diskette functions, you will need to refer to the 4700 Subsystem Operating Procedures manual. To begin the transfer, IPL your 4700 system using the installation diskette. When you receive the display screen that shows the installation menu, enter code 09.

You will select option 2 in the AP Transfer menu to add the applications to the operating diskette. The system monitor will first ensure that sufficient AP space exists. If not, option 1 must be used to allocate more space. If the system monitor indicates you do not have sufficient space, use option 1 with caution. It will erase any existing programs from the SYSAP data set, and you will then have to add not only the applications Application Services diskette, but also any programs that previously existed there.

When option 2 is selected, the system monitor will prompt you to insert the AP diskette (which is the Application Services diskette). If you have only one diskette drive, the system will prompt for the AP diskette and then the operating diskette twice. Each time the system will read an application from the Application Services diskette and write it on the operating diskette. On completion, IPTXFER and IPTXIT will have been transferred to your 4700 operating diskette and entries for them will have been made in the SYSAP data set header.

The following CPGEN macros are also affected:

1. MONOPTS - Required only to do a member upload.

```
MONOPTS
  APMERGE=[N|T|R]
```

2. STATION - Increment your APSTACK operand by 2 if necessary. Code ASYNLDA=0 if the IPTXFER command is sent as keystrokes and ASYNLDA=7 is sent in a data frame.

3. DCA5150 - Code:

```
FEATURE=API
```

4. OPTMOD - The following operands are required on the OPTMOD macro:

```
M45 and P21
```

   The following is required for the /D option on the 4700 SEND command:

```
P41
```

5. TRANPL - If you make IPTXFER transient, use a size value of at least 15.0K bytes:

```
TRANPL    (1,15000)
```

   If you make the IBM-supplied IPTXIT transient, use a size value of at least 1.0K bytes:

```
TRANPL    (1,1000)
```

   If you make both IPTXFER and IPTXIT transient, use a size value of at least 16.0K bytes:

```
TRANPL    (1,16000)
```

   If you code APMERGE=T on the MONOPTS macro, add 3.5K bytes to the size value.

   For example, if both IPTXFER and IPTXIT are transient and APMERGE=T, code:

```
TRANPL    (1,19500)
```

Refer to the *4700 Controller Programming Library, Volume 6: Control Program Generation* for a complete description of any these 4700 CPGEN macros. Also see the *4700 Personal Computer: Addendum to the 4700 Finance Communication System* for more information on file transfer.

The 4700 DCA File-Transfer Program

The 4700 file-transfer program:

- Stores files, sent by the personal computer SEND command, on the 4700 controller's disk.
- Transmits 4700 files requested by the personal computer RECEIVE command.

The DCA file-transfer procedure works like this:

1. To transfer a file, the personal computer acts as a terminal that works with a user-written application program in the 4700 controller. The SEND or RECEIVE commands transfer files.
2. At the personal computer, DOS analyzes the SEND or RECEIVE command and passes control to the file-transfer program (SEND.COM or RECEIVE.COM) supplied with the 4700 Personal Computer Application Services.
3. The personal computer file-transfer program creates a file-transfer command (IPTXFER ...), and transfers that command to the 4700 application program in a DCA data frame, or as a series of keystrokes if the /K option is specified.

4. The 4700 application program, as it analyzes the command from the personal computer, must anticipate and recognize the IPTXFER character string as a command, store the address of the IPTXFER command in a parameter list, and issue APCALL to invoke an IBM-supplied file-transfer processing program.

5. The file-transfer program obtains the address of the IPTXFER file transfer command from the APCALL parameter list, analyzes the command, and accesses the disk or the diskette to store data from a SEND or to obtain data for a RECEIVE.

File-Transfer Command

When the personal computer user enters a SEND or RECEIVE command, the file-transfer command that is sent to the 4700 controller has one of these formats. If the user issues SEND, the personal computer sends:

```
IPTXFER PUT dsname filetype drive (ASCII
        CRLF DCOMP EXIT dssss...d LRECL n
         MEMBER name RECFM n SIZE n
```

If the user issues RECEIVE, the personal computer sends:

```
IPTXFER GET dsname filetype drive (ASCII
        CRLF EXIT dssss...d MEMBER name
```

The 4700 program must recognize the IPTXFER string and store it intact for a later APCALL instruction. So the 4700 application program should read and recognize a character string that begins with the characters "IPTXFER" and ends with the Enter key. The actual fields in the IPTXFER command are parameters used by the file-transfer program supplied by IBM.

The fields in the IPTXFER command are:

dsname The 4700 data-set name. For SEND, if this name matches an existing 4700 data-set name, the file-transfer program replaces the existing data set with the new data set of the same name.

filetype The 4700 data-set organization.

For RECEIVE, this field is ignored.

For SEND, this field indicates the 4700 data-set organization to be used. If this field is missing, ASDS organization is assumed.

ASDS	Arrival sequence data set
ESDS	EDAM sequential data set
EDDS	EDAM direct data set.

drive The 4700 disk or diskette drive.

1	Primary diskette drive
2	Secondary diskette drive
A	Disk drive 1
B	Disk drive 2
C	Disk drive 3
D	Disk drive 4.

Any diskette used must be a standard, labeled diskette with 256 bytes per sector. It must contain a SYSDSLBL data set.

ASCII For SEND, the 4700 program translates ASCII records to EBCDIC before it writes them to the data set.

For RECEIVE, the 4700 program translates EBCDIC records to ASCII before it sends them to the personal computer.

CRLF For SEND, the 4700 program removes carriage-return line-feed characters from each record.

For RECEIVE, the 4700 program adds carriage-return line-feed characters to the end of each record.

DCOMP Specifies that a compress diskette is to be performed when MEMBER is specified and when the SYSAP data set is not large enough to contain the new member.

The P41 optional module must be loaded.

EXIT dsss....d

A literal parameter string to be passed to the user-exit program, IPTXIT. Replace each 'd' with any valid character to be used as a delimiter. Replace 'ssss' with up to 40 characters (not including the delimiters) to send to IPTXIT as a parameter string.

IPTXFER recognizes the first delimiter, and then removes it from the string. IPTXFER sends to IPTXIT all subsequent characters up to, but not including, the next delimiter character. The delimiters are sent to IPTXFER, but they are removed before the parameter string is sent to the IPTXIT exit processor.

LRECL n The character n is replaced by the logical record length of the 4700 data set.

MEMBER name

The member name of a member in the SYSAP data set.

RECFM F or V

Use V for variable-length records or F for fixed-length records.

SIZE n For SEND, the n is replaced by the number of K bytes (K=1024) that must be allocated to the 4700 data set to contain this file. This data is required when you send a file to the 4700 disk; it is ignored when you send data to the 4700 diskette.

See the *IBM 4700 Personal Computer: Addendum to the 4700 Finance Communication System* for additional information on programming to use the 4700 file transfer functions.

Glossary

This glossary explains terms and definitions used in this publication. Some of the definitions may be from the book *IBM Vocabulary for Data Processing, Telecommunications, and Office Systems*.

alphameric. A synonym for alphanumeric, pertaining to a character set that contains letters, digits, and other characters, such as punctuation marks.

argument. A program variable used for passing values or data between processes.

Basic I/O System. The part of an operating system that controls low-level communication with various peripheral devices.

BIOS. Basic I/O System.

block-type device driver. In DOS, a device driver that transfers data in fixed-length blocks (normally a disk/diskette device driver).

control program. A program that supplements the operating system of the personal computer.

critical error handler. A routine that DOS calls automatically if an error occurs in a DOS function call. There is a standard DOS error handler or the user can provide one for special functions.

customization. The specification of a set of operational characteristics.

data set. A 4700 controller disk or diskette file. The personal computer running Application Services can access the contents of special data sets as if they were disks or diskettes.

DCA. Device cluster adapter.

device cluster adapter. A type of interconnection used in the 4700 system. For local communication, you connect a coaxial cable from a personal computer's 3278/79 Emulation Adapter to the 4700 controller's DCA port.

directory. In DOS, a file that can contain files. This allows files to be grouped in subsets on disks or diskettes. The [path] specifier allows access to directory contents.

disk. A fixed disk.

diskette. A flexible (floppy) disk.

disk operating system. A program that interacts with the processor and the disk or diskette to control the flow of data in the computer.

DOS. Disk operating system. Usually means IBM PCDOS.

emulation. The simulation or imitation of one device by another.

extended ASCII. A set of ASCII codes that uses the eighth (most significant) bit to define 127 additional codes. Standard ASCII only uses 7 bits and only defines 128 codes.

Hot-Key. A keyboard key that causes switching between a personal computer application and terminal emulation (mode switching).

Initial program load. The process that loads and executes DOS.

IPL. Initial program load.

mapping. The assignment of one set of data to another, as in the transfer of a set of variables to a set of processor registers or vice versa.

mode switching. Operator switching between a concurrently running personal computer application and terminal emulation.

operating system. The program that resides in a computer to schedule and supervise the running of other programs.

operational diskette. A diskette containing all of the essential files and programs needed to perform a certain task or set of tasks.

port. A point of access or exit. A personal computer port is that point at which a device-connection cable connects to the system unit.

public volume manager. The user of a personal computer that has unrestricted use of the VVOL command. The VVOL command allows a personal computer to manipulate data sets in the 4700 controller.

remote reset volume. A file on the 4700 controller's disk or diskette that is used to supply system data (an IPL diskette image) to attached personal computers.

scan code. A code generated by a keyboard.

system diskette. The diskette, either real or virtual, that contains your control program. For the personal computer, this is the diskette on which you have DOS and the 4700 Personal Computer Application Services.

terminal mode. A single-session, 3278-emulation use of the 3278/79 Emulation Adapter.

translation. Conversion of a code or codes to another code or codes according to a set of specifications.

virtual volume. A data set on the 4700 controller's fixed disk or diskette that acts like a personal computer disk or diskette file.

VVOL.COM. The virtual volume management program supplied with the 4700 Personal Computer Application Services.

VVOL.PUBLIC. A data set in the 4700 controller that contains files and programs that any attached personal computer can access as if it were a personal computer disk or diskette file.

VVOL.SYSTEM. A data set in the 4700 controller that contains the files and programs required by remote IPL. This data set appears to be the IPL diskette to a personal computer performing a remote IPL.

3278/79 Emulation Adapter. In the personal computer, an adapter card that provides 3278-like coaxial communication to the DCA port on a 4701 controller.

Index

E

F

H

I

K

M

O

P

Q

R

S

The Personal Computer
Software Library

Reader's Comment Form

Application Services SC31-3016-1

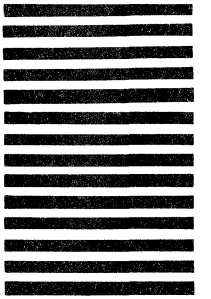
Your comments assist us in improving the usefulness of our publication; they are an important part of the input used for revisions.

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Please do not use this form for technical questions regarding the IBM Personal Computer or programs for the IBM Personal Computer, or for requests for additional publications; this only delays the response. Instead, direct your inquiries or request to your authorized IBM Personal Computer dealer.

Comments:

NO POSTAGE
NECESSRY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK 10504

POSTAGE WILL BE PAID BY ADDRESSEE

IBM CORPORATION
DEPARTMENT 78C
1001 W.T. HARRIS BOULEVARD
CHARLOTTE, NC, USA 28257

Fold here

Tape Please do not staple Tape

The Personal Computer
Software Library

Reader's Comment Form

Application Services SC31-3016-1

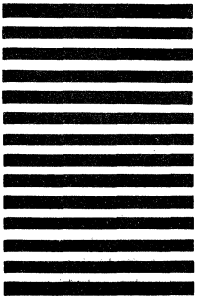
Your comments assist us in improving the usefulness of our publication; they are an important part of the input used for revisions.

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Please do not use this form for technical questions regarding the IBM Personal Computer or programs for the IBM Personal Computer, or for requests for additional publications; this only delays the response. Instead, direct your inquiries or request to your authorized IBM Personal Computer dealer.

Comments:

NO POSTAGE
NECESSRY
IF MAILED
IN THE
UNITED STATES



BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, NEW YORK 10504

POSTAGE WILL BE PAID BY ADDRESSEE

IBM CORPORATION
DEPARTMENT 78C
1001 W.T. HARRIS BOULEVARD
CHARLOTTE, NC, USA 28257

Fold here

Tape

Please do not staple

Tape

International Business
Machines Corporation
1001 W.T. Harris Blvd.
Charlotte, NC 28257

Printed in the
United States of America

SC31-3016-1