IBM System/3
Model 15
Communications Control Program
System Reference Manual

Program Numbers:

5704-SC1

5704-SC2

Features 6011/6012/6033/6070/6071

GC21-7620-6
File No. S3-36

Seventh Edition (September 1980)

This is a major revision of, and obsoletes, GC21-7620-5. Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change or addition.

This edition applies to the following IBM System/3 system control programs and to all subsequent versions and modifications until otherwise indicated in new editions or technical newsletters.

Version	Modification	Program Number	System/3 Model
08	00	5704-SC1	Model 15 A-B-C
05	00	5704-SC2	Model 15D

Changes are periodically made to the information herein; changes will be reported in technical newsletters or in new editions of this publication.

Use this publication only for the purposes stated in the *Preface.*

Publications are not stocked at the address below. Requests for copies of IBM publications and for technical information about the system should be made to your IBM representative or to the IBM branch office serving your locality.

This publication could contain technical inaccuracies or typographical errors. Address your comments about this publication to IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901. 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.

This publication describes the Communications Control Program (CCP) feature of the IBM System/3 Model 15 and provides information to aid the system installation manager, IBM systems engineer, and application programmers in generating the CCP system.

The generation of the CCP system is accomplished by using the facilities of the:

- IBM System/3 Model 15 System Control Programming (5704-SC1 and 5704-SC2)
- Programming support for the communication adapters: Multiline/Multipoint feature Multiple Line Terminal Adapter feature (PSHRPQ number 5799-WFK)

Prerequisite Knowledge

You should be an experienced System/3 Model 15 user familiar with the basic concepts of teleprocessing.

Devices and Programs Supported and Required

The terminal devices, system devices, and system programs required and supported by the CCP are listed in Appendix D.

Note: This manual follows the convention that *he* means *he* or *she.*

Related Publications

- *IBM System/3 Communications Control Program Messages Manual,* GC21-5170.
- *IBM System/3 Model 15 Communication Control Program System Operator Guide,* GC21-7619.
- *IBM System/3 Communication Control Program Terminal Operator's Guide,* GC21-7580.
- *IBM System/3 Communications Control Program Programmer's Reference Manual,* GC21-7579.
- *IBM System/3 Communications Control Program System Design Guide,* GC21-5165.
- *IBM System/3 Model 15 Operator's Guide,* GC21-5075.
- *IBM System/3 Model 15 System Control Programming Reference Manual,* GC21-5077 (5704-SC1).
- *IBM System/3 Model 15 System Control Programming Concepts and Reference Manual,* GC21-5162 (5704-SC2).
- *IBM System/3 Model 15 System Control Programming Macros Reference Manual,* GC21-7608.
- *IBM System/3 Overlay Linkage Editor Reference Manual,* GC21-7561.
- *IBM System/3 Model 15 Scheduler Logic Manual,* SY21-0035.
- *IBM System/3 Model 15 Supervisor and IOS Logic Manual,* SY21-0033.
- *IBM System/3 Model 15 Data Management Logic Manual,* SY21-0034.
- *IBM System/3 Model 15 System Data Areas and Diagnostic Aids,* SY21-0032 (5704-SC1).
- *IBM System/3 Model 15 System Data Areas and Diagnostic Aids,* SY21-0052 (5704-SC2).
- *IBM System/3 Model 15 System Services Programs Logic Manual,* SY21-0036.
- *IBM System/3 Multiline/Multipoint Binary Synchronous Communications Reference Manual,* GC21-7573.
- *IBM System/3 Multiple Line Terminal Adapter RPQ Program Reference and Component Description Manual,* GC21-7560.
- *IBM 1050 Data Communication System Principles of operation,* GA24-3474.
- *IBM 2740 Communications Terminal Models 1 and 2 Component Description,* GA24-3403.
- *IBM 2741 Communication Terminal,* GA24-3415.
- *IBM System/7 Systems Summary,* GA34-0002.

- *IBM System/7 Binary Synchronous Communications Module (RPQ) Programming Guide and Reference Manual,* SC34-1510.
- *IBM System/7 Teleprocessing Multiplexor "TPMM" Programming Guide and Reference Manual Supporting RPQ D08011,* SC34-1506.
- *IBM 3270 Information Display System Component Description,* GA27-2749.
- *IBM 3735 Programmer's Guide,* GC30-3001.
- *IBM System/3 3735 Support Program Coding Manual,* GC21-5096.
- *IBM 3741 Data Station Reference Manual,* GA21-9183.
- *IBM System/3 Disk Sort Reference Manual,* SC21-7522.
- *IBM System/3 Model 15 User's Guide to Spooling,* GC21-7632.
- *IBM System/3 Subset American National Standard COBOL Reference Manual,* GC28-6452.
- *IBM System/3 RPG II Reference Manual,* SC21-7504.
- *IBM System/3 FORTRAN IV Reference Manual,* SC28-6874.
- *IBM System/3 Basic Assembler Program Reference Manual,* SC21-7509.
- *IBM System/34 Interactive Communications Feature Reference Manual,* SC21-7751.

LIST OF ABBREVIATIONS

ASCII	American National Standard Code for Information Interchange
BSC	Binary synchronous communication
BSCA	Binary synchronous communications adapter
BSCC	Binary synchronous communications controller
CCP	Communications control program
CPU	Processing unit
CRT	Cathode ray tube
DA	Display adapter
DFCR	Display format control routine
DFF	Display format facility
DFGR	Display format generation routine
DSM	Disk system management
EBCDIC	Extended Binary Coded Decimal Interchange Code
EOJ	End of job
EOT	End of transmission
EXIO	Execute input/output
I/O	Input/output
ICF	Interactive Communications Feature
IOCS	Input/output control system
IOS	Input/output supervisor
IPL	Initial program load
LRC	Longitudinal redundancy check
MFCM	Multi-function card machine
MFCU	Multi-function card unit
MLMP	Multiline/multipoint
MLTA	Multiple line terminal adapter
MRT	Multiple requesting terminal
NEP	Never-ending program
OCL	Operation control language
PAS	Program appended storage
PCT	Program control table
PFGR	Printer format generator routine
PRUF	Program request under format
RPQ	Request for price quotation
SCP	System control program
SWA	Scheduler work area
TP	Teleprocessing
VTOC	Volume table of contents

Chapter 1. Introduction

The communications control program (CCP) is a system control programming feature that allows the IBM System/3 Model 15 to support an online network of terminals. CCP enables terminals to call application programs as needed and permits those programs to access a common set of disk files. If sufficient main storage is available, the CCP permits several application programs to be executing concurrently, though independently of one another; that is the CCP provides for multiprogramming within one partition.

The CCP is designed to make a communications-based system as easy and inexpensive as possible to establish and operate. The CCP can be tailored to suit diverse data processing environments involving batch and online applications.

Note: If you are not acquainted with the terms used in this introduction, you can find them explained either in Appendix E: *Glossary,* or in the *Data Processing Glossary,* GC20-1699.

TERMINAL OPERATOR FACILITIES

Under control of the CCP, the operator of a terminal can:

- Request programs.
- Specify whether a program request should be rejected if the program cannot be executed immediately, or whether the request should be placed on a queue.
- Specify the disk files to be used by a particular program or series of programs.
- Change the symbolic name of the terminal.
- Send a message to the system operator.
- Release the terminal from a program in order to enter another program request or command.
- Request the operation of a series of tasks via task chaining (Program Number 5704-SC2 only).
- Request sort operations via the CCP/Disk Sort Program. (Program Number 5799-ATH is used with SCP 5704-SC1; 5704-SM7 is used with SCP 5704-SC2.)
- Find new or updated programs and formats (Program Number 5704-SC2 only).
- Request the online test to determine whether the terminal is operating correctly.

SYSTEM OPERATOR FACILITIES

The system operator initiates and terminates the activity of the CCP and controls the operation of the communications-based system. After the CCP is loaded into main storage, it asks the system operator one or more questions. This allows the system operator to identify the set of files, programs, terminals, communication lines, and terminal names to be used by the CCP on the current run (one or more of these sets were defined prior to the current run of the CCP—see index entry: *assignment stage*). These questions also allow the system operator to modify a selected set to adapt to a particular run of the CCP.

During the operation of the CCP, the system operator controls the system through the Console Keyboard. The operator can:

- Monitor the status of the system at any moment.
- Determine the unfulfilled requests for programs or system operator replies in the system at any time.
- Send messages to terminals.
- Change the status of terminals on the system.
- Cancel, suspend, and resume activities of programs.
- Change the actual terminal referred to by a terminal name.
- Request the online test of terminals to determine whether they are operating correctly.
- Close CCP files.
- Open the previously closed CCP files.
- Shut down CCP.
- Change interval polling times.
- Recover from terminal errors.
- Save trace table information.
- Activate or deactivate automatic retries to terminals in error recovery and control the frequency of retries (Program Number 5704-SC2 only).
- Initiate or terminate the CCP loadable trace.

The system operator can also perform some of the functions of a terminal operator such as request programs, enter commands, and, for Program Number 5704-SC2, find programs and formats.

Because of the extent of the control exercised by the system operator, he or she must be thoroughly trained in the operation of the CCP, applications of the CCP in the installation, and the specific tasks to perform under control of the CCP.

PROGRAMMING FACILITIES

Programs that run under the CCP can be written in any of four programming languages:

- RPG II
- COBOL
- FORTRAN IV
- Basic Assembler

Although the design of programs written for the CCP might be different from those the programmer has been writing, programming statements used for terminal input/output are already familiar to programmers:

- In COBOL or FORTRAN: the call statement.
- In RPG II: either the exit operation or a special file.
- In Basic Assembler: macro instructions that can be processed by the disk system management macros.

With each request for terminal I/O, the programmer provides a list of parameters that tell the CCP which specific operation to perform, which terminal to use, and what data area to use.

The CCP allows the programmer to identify terminals by symbolic names. If a particular terminal is unavailable for any reason, the system operator can reassign the symbolic name to a different terminal. Thus, the program need not be changed or recompiled to address a different terminal.

Other facilities offered by the CCP to the programmer are:

- Access to the name of the terminal that requested the program
- Access to attributes of individual terminals
- Support for overlay programs
- Automatic translation of transmission data codes
- Dynamic, program-controlled allocation and deallocation of terminals
- Access to communications I/O error or exception information
- Interaction with external buffers (Program Number 5704-SC2 only)
- Initiation of a task (including a sort task) from an active task via task chaining (Program Number 5704-SC2 only)

DEVICES AND PROGRAMS SUPPORTED

The terminal devices, system devices, and system programs required and supported by the CCP are listed in Appendix D.

ESTABLISHING AND OPERATING THE CCP

The CCP can be tailored to suit each unique operating environment. Establishing and operating the CCP in a particular environment is accomplished in three stages:

- Generation
- Assignment
- Operational system startup

Generation Stage

CCP generation is the process by which a user creates an individual version of the CCP. The purpose of generation is to create a set of CCP object modules and subroutines, unique to that user's requirements, on the user's disk pack. The process of generation involves:

1. Describing the type of equipment to be used by the communications system and the facilities that should be included in that system.
2. Creating a set of control routines whose specific content may be unique to the user's installation.
3. Joining the routines by a link-editing process.
4. Copying appropriate additional supporting routines.
5. Initializing the assignment file used by the assignment stage and the operational stage.

Assignment Stage

Assignment is a special, brief CCP run during which the user specifies one or more sets of specific environments in which the CCP will run. Each set includes:

- Specific items of information pertaining to the entire CCP, such as the current password.
- Programs that can be run under the CCP and the resources each requires.
- Files that are accessible to each program and how they are to be accessed.
- The current line/terminal configuration.

The assignment run need be repeated only when the user wishes to change some of the specific information given in a previous assignment run.

Operational Stage

The operational stage begins with operational startup, when the CCP is loaded into main storage. During startup, the CCP routines open disk files, adapters, and communication lines and complete various tables and control blocks.

During operation, the CCP performs the functions requested by terminal operators and the system operator, executing application programs as directed by those operators.

The operational stage is concluded by shutdown, which is initiated by the system operator. During shutdown, the CCP allows currently executing programs to complete processing, then closes communication lines, adapters, and files.

CCP TASKS

Each body of code, such as an application program, executing independently in the CCP program partition is considered a *task*. It is possible for two or more copies of the same program to be executing concurrently and independently under the CCP; in this case, each copy is considered a task.

Tasks are identified as either user tasks or system tasks (see *CCP System Operator's Guide*, for uses of the task identification). An example of a user task is an inquiry program, loaded into main storage by request from either a terminal operator or the system operator; an example of a system task is the CCP communications management task, which processes requests for terminal I/O.

The management of tasks, as performed by the CCP, permits tasks to be initiated and terminated independently and to operate concurrently. When user tasks run concurrently, results are identical to those that would occur had each task run alone. Under CCP, 1-15 user tasks can execute concurrently; tasks can be from 4K to 32K bytes.

RELATIONSHIP TO OTHER PROGRAMS

Figure 1 illustrates the relationships between the CCP and disk system management and between the CCP and application programs.

Disk System Management (DSM)

The CCP operates in conjunction with DSM and uses the following facilities of DSM wherever possible:

- *DSM Supervisor:* Used for program loading, disk and unit record physical I/O control (IOS). On Program Number 5704-SC2, use enqueue and dequeue.
- *Disk and Unit Record Data Management:* Used for logical control of disk and unit record I/O.
- *DSM Console Management:* Certain CCP display transients run in the DSM transient area and interface directly with DSM console management transients.
- *MLTA and/or MLMP IOCS:* The CCP incorporates either or both of these System/3 communications I/O programs, depending upon the line configuration of the system: the multiple line terminal adapter (MLTA) RPQ (request for price quotation) program for start-stop terminal; the multiline/multipoint (MLMP) IOCS for binary synchronous terminals attached to System/3 by the binary synchronous communications adapter (BSCA lines 1 and 2). Your communications programs do not (and *must* not) use the communications IOCS directly; the CCP calls the appropriate IOCS when needed, based on the instructions you issue in your program.
- *BSCC IOCS:* The CCP incorporates this I/O support if BSC lines 3 or 4 are used. This I/O support is part of DSM and can be used if BSCC was specified during system generation. This support is used only by CCP.
- SIOC IOCS: The CCP incorporates this I/O support if an RPQ (request for price quotation) for SIOC channel connected systems is used. For additional information, see *IBM System/3 Model 15D Channel Connected Systems Program Reference and Logic Manual,* GC21-5199 (Program Number 5704-SC2 only).

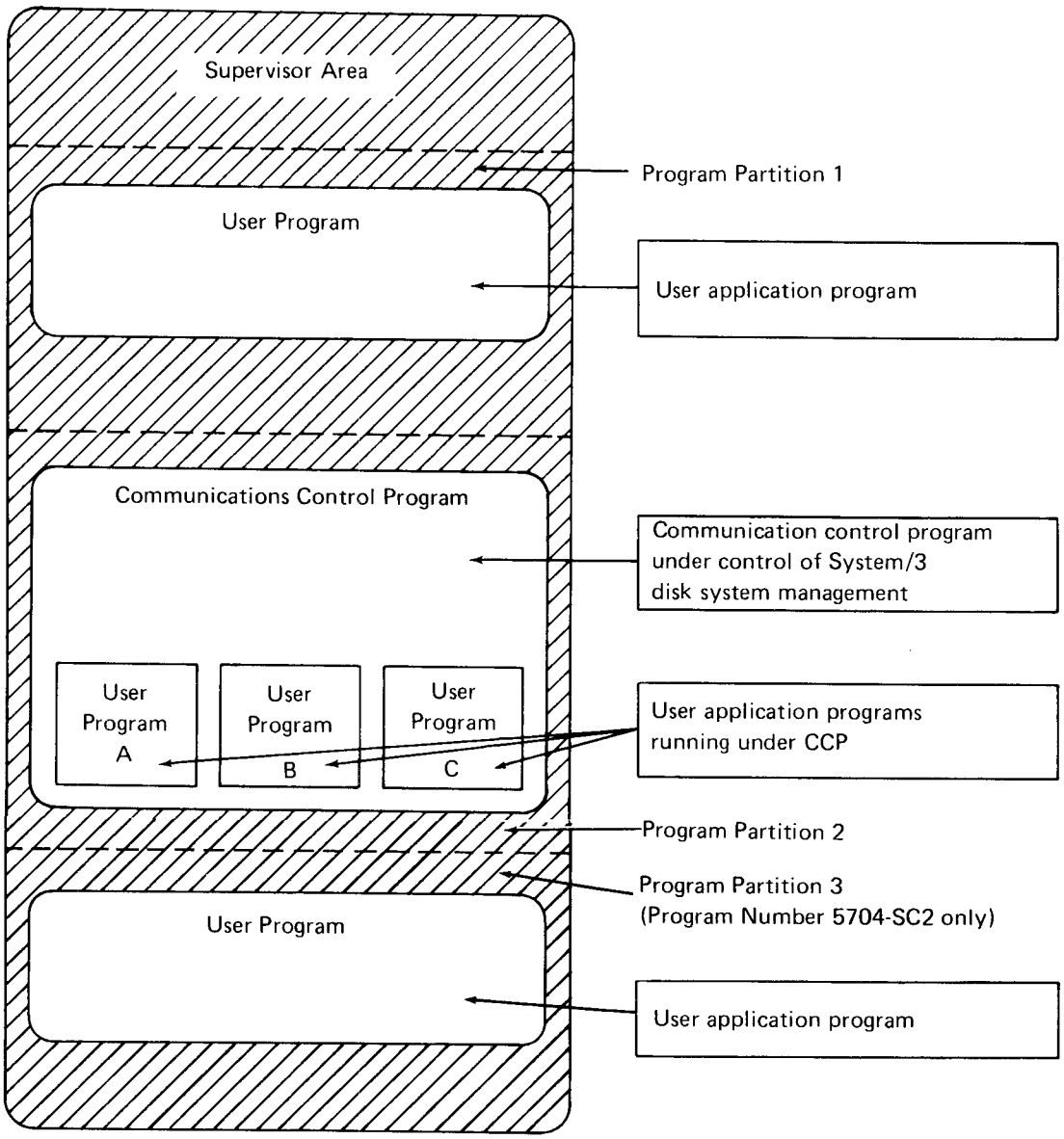


Figure 1. Relationships between System/3 Disk System Management, the Communications Control Program, and User Application Programs

Existing Communications Programming Support

The RPG II telecommunications feature, which provides support for batch terminals in RPG II, is not a part of the CCP and cannot be used under the CCP. Application programs using these features can, however, be executed in another program partition from the CCP as long as their use of communication lines does not conflict with the CCP.

Application Programs

Application programs can execute under control of the CCP or, in another partition, under control of DSM.

Telecommunication application programs that execute under the CCP are one step removed from control by DSM. They are loaded by the CCP and receive control from the CCP. Requests by these programs for system services are received and scheduled by the CCP. Some requests are performed by the CCP; some are passed from the CCP to DSM to be performed.

A telecommunications application program that executes in a non-CCP partition must include the communications IOCS routines it requires. Its use of communications lines must not conflict with the CCP.

Some noncommunications application programs not originally written to run under CCP control can be executed under the CCP. These programs can be requested from the system operator's console or from a terminal if they do not violate any of the restrictions placed upon CCP programs and if they are defined in an assignment set. If such a program has not been rewritten to release the requesting terminal, then the terminal is not free to enter another command or program request until the requested program terminates.

Noncommunications application programs can be executed in a non-CCP program partition while the CCP is operating if they do not conflict with the CCP in their use of system resources (disk files and unit record devices).

System Programs

System program facilities[1] such as utilities, compilers, source library Get/Put, or SYSPRINT) cannot be run under CCP. Some system programs can be run in the non-CCP partition(s) while CCP is executing, and some dedicated system programs require that CCP be shut down before they can be executed. For more information, see the appropriate SCP reference manual listed under *Related Publications* in the Preface.

[1] The spool file copy program ($QCOPY) is an exception. $QCOPY runs under control of SCP and, if its name is changed so that it begins with something other than $, $QCOPY also runs under control of CCP (Model 15 5704-SC2 only).

To the operator of a terminal, the system is a resource to help him accomplish his tasks. The terminal operator might think of the system as belonging to him or her alone, unless the combined demands upon the system are great enough to cause delay in its responsiveness. All contention among terminal operators for use of the system is managed by CCP.

The CCP distinguishes two types of terminals:

Command terminals, which can request services of the CCP including the running of application programs.

Data terminals, which cannot request services, and are used only as directed by the application programs.

In order to call for application programs, the command terminal operator must first sign on to the CCP. A sign-on request is a message initiated by the terminal operator signifying the wish to request services of the CCP. If the system has a password security feature (an option selected during the CCP generation—see index entry *initial mode*), the terminal operator must correctly enter a password with the sign-on request.

Once a terminal operator is in communication with the requested application program, he or she enters data as required by the program. The sequence of operations at the terminal, and the format of data sent to and from it, are entirely directed by the application program. The terminal operation continues to be directed by the application program until that program releases it. However, the operator, while sending data from the terminal to the application program, can reestablish communication with the CCP in order to:

- Send a message to the system operator and then resume sending data to the program.
- Release his terminal from control of the program.

When a terminal operator finishes making a series of requests for services of the CCP, he or she normally signs off the terminal. This action restores the terminal to an initial status, such that it must be signed on again (with a password, if that option was chosen) before it can request services of the CCP.

HOW THE TERMINAL OPERATOR REQUESTS A PROGRAM

When a command terminal is not in use, it is continually monitored by the control program for a request. The terminal operator calls for an application program to perform a specific function by simply entering the program name at the terminal. The CCP then attempts to load and execute the program and to put that program in communication with the terminal operator. From then until the completion of the program, the interaction between the terminal operator and the system is dictated by that program.

When the application program completes execution, it yields control of the terminal to the CCP. The CCP once again monitors the terminal for a terminal command such as a program request. The next program request does not have to be for the same program.

Each terminal operator must be trained in the functions he or she can call upon and in the procedures for interacting with the application programs that perform those functions.

HOW THE TERMINAL OPERATOR REQUESTS SYSTEM SERVICES

While monitoring for program requests, the CCP can also respond to commands to perform services for the terminal.

The operator can specify ahead of time what the system's response should be when it is unable to comply immediately with his request for a program. The system might be temporarily too busy with requests from other operators. By command, he can choose one of the following responses:

- The system denies the request and allows him to make some other request.
- The system holds the request and honors it as soon as possible.

Once the operator of a terminal has specified one of these requests, the CCP handles all program requests from that terminal accordingly until the operator specifies differently.

A file specification allows the operator to specify the disk data files that are to be accessed by programs he or she requests, if those programs are written to accept the specification. Within the information system, there may be several files containing similar data in the same format. A school system, for example, might have a separate student records file for each school. An application program requested by the operator might have been written to access any one of these files, but the program must be told which file to use on a particular run. A file specification command issued by the terminal operator applies to all programs requested at that terminal until a contradictory command is issued.

Another command permits the terminal operator to send a message to the system operator, requesting some action to be taken.

The commands discussed so far are issued to the CCP while it is monitoring a terminal for requests. Once a terminal is interacting with an application program, however, the input from that terminal is meaningful only to the application program, with one exception. The CCP checks each message from a terminal to a program for a certain string of characters defined in your system. When it detects these characters in a message, the CCP interprets the message as an attempt by the terminal operator to escape from control of the application program and communicate directly with the CCP. At this point, the CCP accepts a request from the operator to release the terminal from the control of the application program, or to send a message to the system operator. If the request was to send a message, the terminal operator can ask the CCP to resume execution of the application program after the message is sent.

TERMINAL MODES

There are two classes of terminals defined previously, based on whether or not the terminals are capable of entering commands to the CCP: command terminals and data terminals (see Appendix E: *Glossary*). Data terminals are capable only of transmitting or receiving data under control of an application program; they are not capable of commanding CCP services. When data terminals are not communicating with an application program, they are in a standby mode (not polled by the CCP for input). Because the operator of a data terminal does not interact with the CCP, this chapter deals only with the operation of command terminals.

Although there are operating differences among the various terminal types that can be used as command terminals, the functions that can be performed by them are the same.

The primary function of any command terminal is to request the execution of application programs. All of the activities a terminal operator performs are related to that function.

Initial Mode

When a command terminal is online, it is physically attached to the system and logically attached to the CCP. The CCP monitors it continuously for program requests or other commands. Before a program can be requested, the operator must first sign on at the terminal. Signing on involves communication between the terminal operator and the CCP. Before and during this communication, the terminal is in *initial mode.* Commands for CCP services other than system operator communication cannot be issued from a terminal when it is in initial mode. The end of initial mode occurs when the terminal operator is successfully signed on.

From the point of view of the terminal operator, signing on may be as simple as entering the sign-on command. However, if access to the system from a terminal must be limited to certain authorized people, the sign-on procedure can involve providing additional information required by a security feature. The security feature can be either the password security option provided by the CCP (see index entry: *password security option*) or a routine written by the user to control access to the system in some other way.

Command Mode

After a terminal operator is successfully signed on, the terminal is in *command mode.* This means the operator can request the CCP to load and execute programs and can issue related commands. Once a terminal is in command mode, it remains in command mode until a program request is made from the terminal (see *Terminal Operator Commands*), the operator signs off, or the system operator varies the terminal offline.

Data Mode

Once the terminal operator issues a command to load and execute a user application program, the CCP loads the program and gives it control. At that point, the terminal enters *data mode;* that is, the terminal is in communication with the application program itself. The nature of the communication is, of course, determined by the application program. Normally, the terminal remains in data mode until the application program completes processing or releases the terminal. At that time, the terminal is again placed in command mode and is able to issue another program request.

Command Interrupt Mode

The operator of a terminal need not wait until a program completes its job in order to interrupt it. By entering a string of six characters that are significant to the CCP (determined by the user at generation time), he can indicate that he wants to escape from data mode and enter command interrupt mode (see index entry: *data mode escape*). While in this mode, he can send messages to the system operator, resume execution of the program, or release his terminal completely from the control of the program (at which point the terminal is again in command mode).

TERMINAL OPERATOR COMMANDS

Two logical groups of terminal operator commands can be issued after sign-on. First, while the terminal is in command mode, before a program request is actually made, the terminal operator can issue various commands pertaining to the subsequent program request:

Tell the CCP how to handle his request if it cannot be honored immediately (see *Queue/No-Queue Commands*).

Issue commands that indicate which files are to be accessed by program requests (see *File Specification Command*).

Tell the CCP by what name, of a set of names defined as valid, the terminal should be known to the program he is requesting (see *Name Command*).

Send a message to the system operator (see *Message Command*). This can also be done prior to sign-on.

The second group of commands is used during command interrupt mode. After data mode escape, the terminal operator can:

- Send one or more messages to the system operator (by using the message command).
- Release the terminal from control of the application program (see *Release Command*).
- Resume execution of the program (see *Run Command*).

Sign-On Command (/ON)

The sign-on command notifies the CCP that the terminal operator wishes to begin making requests of the system. If the system uses a security feature, the sign-on command must be accompanied by one of the following:

1. The current password required by the CCP password security feature.
2. Information required by a user-written sign-on routine.

The CCP logs every sign-on attempt on the system operator's console and indicates if sign-on was successful. If the sign-on was successful, the CCP notifies the terminal operator and allows him to enter a command. If the sign-on was not successful, the CCP allows the terminal operator to attempt to sign on again.

While a terminal is in initial mode, the name of the terminal can be changed. The system operator can change the name by using the assign command. Sign on restores the original name of the terminal.

Once the operator signs on at the terminal, he can make any number of requests without signing on again. However, if the terminal operator leaves the terminal unattended and access to the terminal is restricted by a security feature, he should sign off when he leaves (see *Sign-Off Command*). If he signs off, he must sign on again when he wants to use the terminal again.

Queue/No-Queue Commands (/Q and /NOQ)

The queue or no-queue command indicates how the CCP is to handle program requests from this terminal which cannot be honored immediately:

Command	Meaning
/Q	The operator waits for the program to start. The CCP places the request on a queue and honors it as soon as possible.
/NOQ	The operator does not wait if the program cannot start immediately. The CCP rejects the command if it cannot be honored immediately and allows the operator to enter another request.

A queue or no-queue command remains in effect until the terminal operator enters a different queue or no-queue command or until he signs off. If neither a queue nor a no-queue command is entered at the terminal, the CCP assumes the no-queue option. For tasks requested via task chaining, if the request was successful, CCP handles the request the same as a queued terminal program request.

Note: Once a terminal has a program request queued, the request cannot be removed from the queue except by the system operator. Queued chain task requests cannot be canceled.

There are five situations where the /Q command is effective for program requests:

- A disk file is temporarily not available because of file sharing.
- The main storage for the program is not available.
- A required terminal is temporarily in use.
- A multiple requesting terminal (MRT) program is active and already has the maximum number of requesters.
- The maximum number of programs might already be executing.

File Specification Command (/FILE)

The file specification command specifies which of several data files to use on a current program run. The terminal operator may use the file specification command to vary the files that are used by the programs he requests. The file specification command cannot be used with multiple requester programs and is not used with a chain task request.

Certain application programs are written to access any of several files containing similar data in the same format by referring to a *symbolic file* in the program. For the program to actually access a file, the name of the symbolic file must be associated with the name of a file that actually exists on disk, a *physical file.*

Suppose, for example, a school system has a separate student records file for each school. A student report program can process the student records file from any of the schools, but it must be told, by a file specification command, which of the files to use on a particular run (Figure 2).

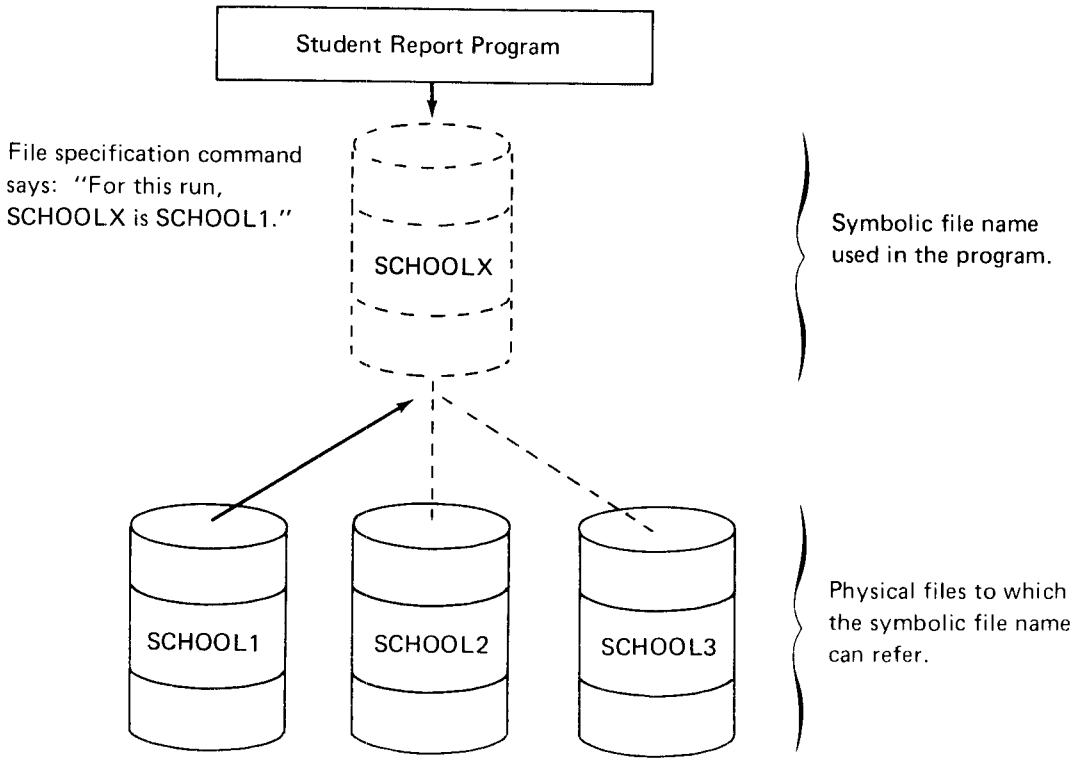


Figure 2. Illustration of File Specification Command

A file specification command is in effect for all subsequent program requests from that terminal until the terminal operator enters a new file command for the symbolic file or until he signs off from the terminal. Thus, the file specification command can apply to more than one program.

If a terminal operator enters a file specification command without naming any files, he cancels all file entries currently maintained for that terminal by the CCP.

If he enters a file specification command that gives only a symbolic file name with no associated physical file name, that symbolic file entry is deleted from the list of file entries maintained by the CCP for that terminal. Signing off from the terminal cancels all file specifications in effect for that terminal.

The CCP informs the terminal operator if the file specification command he entered was invalid. When doing task chaining, the /FILE command is valid only for the first task (the task loaded via a terminal or console). The symbolic file associations are not considered valid for other tasks in the chain.

Name Command (/NAME)

The name command tells the CCP which symbolic name to use for this terminal. This symbolic name is taken from a set of previously assigned symbolic names. The CCP uses this name for this terminal from sign-on to sign-off, or until another name command is entered. The CCP passes the symbolic name to any program this terminal requests in order to identify the requester (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579).

The CCP maintains a list of symbolic names eligible for each terminal. These names are associated with the terminal during the CCP assignment. The terminal to which they apply can be changed by the system operator. The primary purpose of the name command is to allow a terminal to assume (with the consent of the system operator) the name of another terminal (of a similar type) that is currently not operable.

Program Request Command

The terminal operator requests a program to execute by entering the name of the program. If a program allows, input data can be entered with the program request.

Before the operator enters a program request, he should consider whether queue or no-queue, file, and name commands are required for that program. He should determine this from a document such as a program run sheet for that program (see *Planning Considerations*), if he is not familiar with the requirements of the program.

The CCP informs the terminal operator by a message if it rejects the program request. For a chain task request, all messages are directed to the system operator.

Data Mode Escape Command

The terminal operator can interrupt an application program he requested (when the terminal is in data mode, under control of the program). To do this, he enters a predetermined set of six characters when the program attempts to get input from the terminal. The string can consist of any six characters previously defined, during generation, to the CCP as the data mode escape characters. While a requesting terminal is in data mode, the CCP continuously monitors input from the terminal for these characters as the first six input characters.

After the terminal has interrupted the application program, the terminal is in command interrupt mode and the operator can:

- Send one or more messages to the system operator (see *Message Command*).
- Resume execution of the application program (see *Run Command*).
- Release his terminal completely from the application program he interrupted (see *Release Command*).

Release Command (/RELEASE)

The release command can only be issued while the terminal is in command interrupt mode; it is invalid at any other time.

The release command releases the terminal from the control of the application program. The application program then receives a return code. This return code informs the application program of the terminal operator's action. The terminal operator is then free to enter another program request or other commands.

Message Command (/MSG)

The terminal operator uses the message command to send a message to the system operator. The text of the message follows on the same line. The message command can be entered when the terminal is in initial mode, command mode, or command interrupt mode.

Run Command (/RUN)

The run command causes an application program to resume reading input data from a terminal. This happens after the terminal issued a data mode escape command and, possibly sent one or more messages to the system operator. This command returns the terminal to data mode from command interrupt mode.

Sign-Off Command (/OFF)

The sign-off command returns the terminal to initial mode or places it offline. The sign-off command can be accompanied by either the word *hold* or the word *drop.*

Hold — The terminal is returned to initial mode. The CCP accepts only a sign-on command or a message command from the terminal.

Drop — If the terminal is connected by a switched line, the line is disconnected; the terminal can re-establish the connection only by redialing the system. If the terminal is connected by a non-switched line, the terminal is placed offline; that is, the CCP will accept no commands from the terminal until it is again placed online by the system operator (see index entry: *change the status of a terminal*). At that time, it enters initial mode.

If neither hold nor drop is entered, the CCP assumes the option that was selected during the CCP assignment stage (see index entries: *MLTATERM statement and BSCATERM statement*). All sign-off commands are logged on the system operator's console with an indication of whether hold or drop was used as the sign-off option.

The sign-off command can be entered anytime after the sign-on command, while the terminal is in command mode. It cannot be entered following a data mode escape command unless the release command has been issued.

The sign-off command clears all file specification entries in effect for this terminal (see *File Specification Command*) and restores the original name of the terminal (see *Name Command*).

PLANNING CONSIDERATIONS

Because the terminal operator will probably be calling programs he did not write, he must have access to information about the programs (perhaps a program run sheet for each program he can call). This sheet will be unique to each installation, but will probably contain answers to the following questions:

- What, from the terminal operator's point of view, is the function of the program?
- Should the program request be queued if it cannot be honored immediately, or should it be rejected? (Should the terminal operator enter a queue or no-queue command?)
- Does the program have to be told which files to use? (Should the terminal operator enter one or more file specification commands?)
- Does the program expect to be requested by this terminal? (Must the terminal operator enter a name command?)
- Should the terminal operator enter input data for the program at the same time he makes the program request?
- What kind of input (and in what format) does the program expect from the terminal operator?
- Are there any exceptional conditions or special situations that can arise from the program? (Must the terminal operator notify the system operator in certain situations? Does the program diagnose error conditions, what are they, and what must the terminal operator do?)
- What are the names of follow-on programs that might be initiated via task chaining?
- Is there any file contention for files being used (such as NOSHR on Assignment and another partition using file)?

In addition to a program run sheet for each program he can call, the terminal operator also requires some current information about the system, such as:

- If the password security option is in effect, what is the current password?
- If the installation has its own unique sign-on procedure, what is it?
- What is the current set of terminal name assignments for the terminal?
- What are the data mode escape characters for the system?
- Should the terminal operator specify HOLD or DROP when signing off—what is the current default for the terminal? Perhaps the operator should not sign off until the end of the day.

The terminal operator might also require information about the schedule of work he is expected to complete.

- Are there certain programs called each day?
- Must the programs be called in any particular order?
- Is the use of the terminal limited in any way—to certain hours or to a certain amount of time per use, for example?

Chapter 3. System Operator Control of the CCP

The system operator has the role of exercising final control over the communications-based system. He initiates its activity by loading and running the CCP. He determines when the system should refuse to accept new requests from terminal operators. While the system is operating, he may initiate certain system actions, determine the system's status, and alter the set of terminals permitted to access the system. He must also make decisions when exception situations (such as error conditions) are detected by the communications control program or a program running under its control.

Once the CCP is started, all communication between the control program and the system operator is through the console keyboard. Messages from the system are displayed on the CRT, some requiring responses from the operator. His response also is keyed on the console keyboard. At any point during the execution of the communications control program, the operator can command a system action via the console keyboard.

In addition to the operational control of the CCP, the system operator can take part in the generation and assignment stages of the CCP. See Chapter 6: *Generation Stage* and Chapter 7: *Assignment Stage* for the required procedures and statements.

INITIATING CCP

The system operator starts the CCP by entering OCL statements after IPL (initial program load). The OCL statements load the CCP and supply OCL FILE statements for all disk files used by the programs to be run under the CCP.

After the startup routine is loaded, it asks questions of the system operator that allow him to exercise several options.

- Select the assignment set for this run.
- Change the password assigned in the set.
- Suppress access to certain disk data files that are normally accessible to programs in this set.
- Suppress access to certain application programs normally available in this set.
- Suppress the use of certain communication lines in the set.
- Suppress the use of certain DFF external buffers in the set (Program Number 5704-SC2 only).
- Suppress the use of certain terminals in the set.
- Change the main storage allocation for communications buffer area assigned in the set.

Changes specified to a set during startup do not permanently alter the assignments in the set. The changes apply only to the current CCP execution. To reply to prompts issued during startup, the system operator presses the PF12 key, types in the response, and presses ENTER.

SYSTEM OPERATOR COMMANDS

After startup of the CCP, the system operator can interact with the system in the following various ways:

- Communicate with a remote terminal by telling the CCP to send a message to a specific or all available terminals and by receiving messages from the terminals.
- Inquire into the status of the CCP at any time.
- Modify the status of the CCP.
- Function as a remote terminal by entering program requests (see *Using the System Operator's Console as a Requesting Terminal(s)*).
- Open and close CCP files using open and close programs.
- Find formats (Program Number 5704-SC2 only).
- Initiate or terminate the CCP trace facility (Program Number 5704-SC2 only).
- Control automatic retries of terminals during error recovery (Program Number 5704-SC2 only).

The system operator sends messages, makes inquiries, and modifies the status of the CCP by entering commands using the console keyboard. To enter commands, the system operator presses the PF10 key and is prompted.

All commands consist of an operation code and one or more operands. The operation codes can be used in either their full-length versions or in abbreviated versions. In some cases, operands can also be abbreviated. Specific operation codes and operands are given in the *IBM System/3 Model 15 Communications Control Program System Operator's Guide*, GC21-7619.

Message Command (MSG)

The system operator uses the message command to send a message to a specific or all available terminals. The operator designates which terminal(s) is to receive the message. He does this by including: either the symbolic terminal name, the CCP's terminal reference identification with the message text, or the keyword ALL indicating all available terminals to the current assignment set.

The CCP notifies the system operator when the message cannot be transmitted because the terminal is either offline, not connected, not a command terminal, under control of an application program, or not in the system.

Display Terminal Status (D T)

The following information about a specific terminal or about all terminals defined to the CCP can be displayed:

- Symbolic terminal name.
- Identification of the task using the terminal.
- Encoded attributes and status of the terminal. This includes information about the terminal such as whether it is online or offline, signed on, or awaiting some event; the operating mode it is in; or if it is waiting for a TP buffer.

Display Terminal Assignments (D A)

This command causes the CCP to display the symbolic names and reference identification of a specific terminal or of all terminals in the system. The system operator can use this command to display terminal assignments when he wants to enter a command to change the terminal which is actually addressed by a particular symbolic name (see *Assign a Symbolic Name to a Terminal*).

Display Users Command (D U)

This command displays to the CCP systems operator the status of active user tasks or the status of an individual user task. The CCP displays:

- Task-id
- Program name
- Requesting terminal name
- Suspended/active/allocating/terminating/queued indicator
- User program area main storage size
- User program area main storage address
- NO TASKS ACTIVE if system is inactive
- Size of largest free space
- Size of largest available TP buffer segment and size of total available TP buffer

When the task number of an active task is entered, a display appears containing the:

- Names of all attached terminals
- Indication of terminal waiting for TP buffer
- Labels of all allocated files
- Task attributes such as MRT (multiple requesting terminal), SORT, CHAIN, DFF (display format facility), or NEP (never-ending program)
- Unit record devices allocated to the task

Cancel a User Program or the CCP

This command can be used by the system operator to stop processing a particular user program or to immediately stop the CCP. The CCP informs the system operator if he enters an invalid task identification and program name to cancel a user program.

Suspend Requests/Execution/Initiation of User Programs

This command allows the system operator to suspend user activity in any of the following ways:

- Suspend the execution of all user programs currently running under control of the CCP.
- Suspend the execution of a particular user program running under the CCP.
- Stop the initiation of user programs under the CCP.

- Stop the initiation of a particular user program.
- Do not allow additional program requests from requesting terminals.
- Do not allow commands from command terminals.

The CCP informs the system operator if he enters an invalid task identification and program name or if suspension is already in effect for the programs he specified.

A program can be canceled after its operation is suspended. A program can also be canceled while in wait (/Q) state. Queued chain task requests cannot be canceled.

Resume Requests/Execution/Initiation of User Programs

The resume command allows the system operator to resume user program activity after a previous suspension, as follows:

- When execution of all user programs running under the CCP was previously suspended, resume the execution of all programs.
- Resume execution of a previously-suspended program.
- Allow the CCP to resume initiation of pending program requests.
- When program requests from command terminals were previously suspended, allow command terminals to resume program requests.
- Allow command terminals to resume entering commands.

The CCP informs the system operator if he specified an invalid task identification and program name or if the resumed task was never suspended.

Change the Status of a Terminal (VARY)

This command allows the system operator to change the status of a terminal to online or offline. The command may be given for all terminals on a given line or for a specific terminal. The command is not allowed if the terminal(s) is currently allocated to a program or awaiting a queued program request.

The CCP informs the system operator of the status of the terminal following the command.

Close/Open the BSCA Line (CLOSE/OPEN)

This command allows the system operator to close or open a BSCA line under CCP. The close command closes the line under CCP for use in a batch partition. The open command reopens the line for use under CCP.

Note: This command applies only to BSCA line 1 or line 2; it does not apply to BSCC.

Assign a Symbolic Terminal Name to a Terminal (ASSIGN)

This command allows the system operator to assign a symbolic terminal name to a specific terminal. This command is used, for example, to assign an alternate terminal when a particular terminal is inoperative. This command is invalid when:

- The symbolic terminal name is currently being used by a program running under the CCP.
- The symbolic terminal name is the only name assigned to an online terminal that is in command or initial mode, or has a queued program request.
- Changing the assignment would cause a terminal unit allocation conflict with a current request for a never-ending program.

To change the name of a command capable terminal, not in initial mode, the terminal operator must also enter a command (see index entry: */NAME command*).

Online Terminal Test (TEST)

The TEST command allows the system operator to initiate the online test facility for terminals that are supported by either the MLTA, BSCA IOCS or BSCC (see index entry: *online terminal testing*). The operator can specify that either a single or multiple online tests be run for MLTA devices. The CCP informs the system operator either when the command is invalid for any reason or when the test is started successfully. For BSC, online test results are displayed on the console.

Note: The system operator can initiate a BSC online test only to another processing unit or to the BSCC attachment.

Online terminal test can also be initiated directly by the terminal operator if he suspects that his terminal is not operating correctly or if he wishes to test it before he transmits data. The terminal must be capable of input and output. The CCP or the application program are not informed of a test that is initiated by the terminal operator. Certain terminals might have restrictions as to what online tests can be executed from the central system or requested by the terminal operator.

Change the Interval Polling Time (POLTIME)

This command allows the system operator to change:

- The amount of time that CCP polls terminals before waiting.
- The amount of time CCP waits before resuming polling.

This command is effective only if the interval polling option is selected during CCP generation.

Recover from Terminal Errors (ERP)

This command allows the system operator to remove one or all of the terminals on a line from CCP error recovery after the condition that caused the terminal error has been cleared.

Control Automatic Retries of Terminals During Error Recovery (AUTOERP) (Program Number 5704-SC2 Only)

This command allows the system operator to enable or disable the automatic error recovery facility if that facility is specified by the assignment set currently in use. The system operator can also use this command to change the frequency of retries.

Save Trace Table Information (TRACE)

This command allows the system operator to enable or disable the SCP trace facility.

Initiate or Terminate the CCP Trace (TRACE) (Program Number 5704-SC2 Only)

This command allows the system operator to initiate or terminate the CCP trace facility.

USING THE SYSTEM OPERATOR'S CONSOLE AS A REQUESTING TERMINAL

The CCP allows the system operator to request programs using the console in the same way a terminal operator requests programs. The following commands are available to the system operator:

- /Q
- /NOQ
- /FILE
- Program request

The effect of these commands is the same as described for the terminal operator (see index entry: *terminal operator commands*).

Certain restrictions and assumptions apply to using the system operator's console to request programs:

- The console is always assumed to be signed on.
- The console cannot be in data mode or command interrupt mode. All requests by the user program for input data from the console must be made by a put-then-get operation issued to the console.
- The PF9 key on the console keyboard must be pressed to tell the CCP to expect a program request.
- CCP prompts ENTER PGM NAME.
- Before entering the program name, the PF12 key must be pressed.
- The program name (and data, if applicable) must be entered.
- The ENTER key must be pressed.

Closing and Opening Disk Files Under CCP

This facility allows the system operator to logically close a disk file in order to:

- Make the file unavailable to CCP programs by inhibiting program requests that require the file.
- Merge added records with existing records of add files for subsequent retrieval or updating by CCP programs or by the other partition, then re-open the disk file.

The OPEN or CLOSE request is entered as a program request from the system operator's console. For information on how to use OPEN and CLOSE, see the *IBM System/3 Model 15 Communications Control Program System Operator's Guide,* GC21-7619.

Closing and Opening Disk Files From a Batch Partition (Program Number 5704-SC2 Only)

The $CCPCO utility allows the system operator to logically close a main data area disk file from a batch partition in order to:

- Make the file unavailable to CCP programs by inhibiting program requests that require that file.
- Merge added records with existing records of update files.
- Perform file maintenance such as enlarging, reorganizing, or deleting. This file can then be reopened to continue processing in CCP.

The utility ($CCPCO) is entered by the system operator from an available batch partition. For information on how to use $CCPCO see the *IBM System/3 Model 15 Communications Control Program System Operator's Guide,* GC21-7619.

SHUTTING DOWN CCP

When it is time for the system operator to stop the CCP, he issues the command, SHUTDOWN. The shutdown command allows all programs that are currently running under the CCP to continue until they terminate themselves. Following a shutdown command, the CCP informs the user program, via a return code, that the system operator wants the program to terminate as soon as possible. Each user program executing at that time is responsible for recognizing the return code and terminating.

Once shutdown has started, only commands issued by the system operator are accepted. No new program requests are accepted. However, programs currently being executed or queued for execution run until they terminate. Also task chaining sequences run until completed. The system operator can shut down the CCP without allowing all programs to continue to completion by issuing a cancel command (see *Cancel a User Program or the CCP).*

The CCP will not go to end of job until all executing programs have gone to end of job or have been canceled.

If operating under control of 5704-SC2, the system operator can enter a delay time with the SHUTDOWN command. When a delay time is entered with the shutdown request, the terminal operators online are informed that shutdown is pending, but actual shutdown does not begin until the delay time has elapsed.

SYSTEM OPERATOR MESSAGES

The system operator can receive messages from several sources while monitoring the CCP operation. Some of the messages require a response (see *IBM System/3 Communications Control Program Messages Manual,* GC21-5170).

CCP Responses

One type of message the system operator can receive is a CCP response to a command he has issued (that inquires about or modifies the status of the CCP). These responses can be either confirmations from the CCP that it has carried out a command or error messages when the CCP encounters an error in the command. These messages contain the identification of the task that issued the message.

Messages from Terminal Operators

The system operator can also receive messages sent by a terminal operator using the message command. These messages are accompanied by the reference ID of the terminal that issued the message. These messages might ask for the system operator to return a message. If a response is required, the system operator also issues a message command, accompanied by the symbolic name or CCP reference identification of the terminal that is to receive the message.

Messages from User Programs

The system operator can also receive messages from user programs. To respond to a message from a user program, the system operator enters the task identification given in the output message and the appropriate text.

When a user program goes to EOJ, (end of job) all messages that the user program has written to the system console are removed from the screen. This might cause the message to disappear from the screen before the system operator notices it or is finished with it. In this case, the system operator can refer to the history file or a log to review all the messages.

PLANNING AND THE SYSTEM OPERATOR

The CCP system operator requires a deeper understanding of the system than the operator of a batch system. The operator must make decisions alone and in a variety of situations. Many of these decisions require a thorough understanding of the CCP operation. The system operator can display and modify the current status of the CCP, and must thoroughly understand the effect of actions at the console on the CCP and on the information processing system as a whole.

The system operator should be involved as early as possible in planning for installation of the CCP. Before operating the system, the system operator should become familiar with:

- The functions of the application programs in the system.
- The files used by each program.
- The configuration of the system.
- The current status of the system.
- The current system assignments.

To keep the communication-based system running smoothly when the system operator is absent, a backup system operator should be trained. This might be the system manager or one of the terminal operators or programmers.

OPERATING AIDS

The system operator must have certain current information about the system available at all times. This includes such things as descriptions of CCP programs and current system assignments.

A program description sheet for each application program should include at least the following information:

- Symbolic name of the program.
- Function of the program (including how it affects the files it uses).
- System resources used by the program. This should list the files used by the program, the terminals used, how the terminals are used, and what names the program uses for the terminals. The main storage requirement of the program and its typical operating time should also be given.
- Does the program require one or more file statements?
- What kind of input does the program expect—will input data be required at the same time the program request is made?
- Are there any special considerations? Is the use of the program restricted in any way? Are there potential problems involved in suspending or canceling the program before it has finished its run?
- Is the program a sort program? If so, will it prevent other programs from using the files for a long while?
- What task chain sequences are defined and what resources are required for successful execution of the entire sequence?

In addition to program definitions, the system operator needs other information about the system. He should have a copy of the current CCP assignments with information about the terminals attached to the system, the lines available on the system, the files available, and the programs used under the CCP.

The system operator also needs current system information like:

- What is the current password (or other security information, if the installation has its own security procedures)?
- What are the current data mode escape characters for the system?
- Is there a particular schedule of work to be completed?

Chapter 4. CCP Services for the Application Programmer

The communications control program aids the programmer in two primary ways:

- By relieving him of many programming concerns inherent in an event-driven system, it lets him concentrate on application programs that do the processing he requires.
- It permits him to write application programs that include communications input/output in a high-level language.

Programs that run under the communication control program can be written in any of four languages:

- RPG II
- COBOL
- FORTRAN IV
- Basic Assembler

By writing in RPG II, COBOL, or FORTRAN, the programmer can avoid the strict rules required when using Basic Assembler language.

In whatever language he writes, the programmer can ignore the problems that arise from his program contending with others for system resources. Those problems are managed by the CCP. The programmer is assured that all required resources are available to his program each time it is executed. If necessary, the CCP defers the execution of his program until those resources are available. If his program shares access to a disk data file with another executing program, the CCP and DSM manage the contention. It also prevents the two programs causing a wrong record update through conflicting reads and writes.

Programs that execute under the CCP are written much the same as programs in the same language for a system without telecommunications. That is, the statements used to process data and the handling of data files are identical.

The standard disk data management methods are supported by the control program. Only two elements are likely to differ significantly:

- The overall logic of the program.
- The means of communicating with terminals or with the system operator.

Except for RPG II, the high-level languages do not offer any statements for accomplishing terminal input/output. Terminals cannot be treated as data files. Furthermore, the MLTA IOCS, BSCA IOCS, and BSCC IOCS do not permit access to their facilities directly from a high-level programming language. The communications control program incorporates MLTA IOCS, BSCC IOCS, and/or BSCA IOCS and offers the application programmer a method of using these IOCS facilities to interact with terminals.

The application program indicates terminal actions to the CCP by one of the following statements:

- In COBOL or FORTRAN—call statement
- In RPG II—exit statement, or a special file
- In Basic Assembler—a supplied macro instruction

Each of these statements is accompanied by parameters to indicate the specifics for the operator. See *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579.

Most communication by programs running under the CCP is with the requesting terminal. But the ability also exists to address other terminals. The user chooses a name to address a terminal in the program. This name normally applies to one particular terminal. However, if a certain terminal becomes unavailable during a run of the CCP, the system operator can reassign the name to another terminal. Any program addressing a terminal by the reassigned name addresses the new terminal. For the most part, the application program need not be concerned with the type of terminal with which it is communicating.

An application program communicates with the system operator by addressing the system operator's console as another terminal. A name (CONSOL) is always used to address the console. The only operations available here are:

- Write a message.
- Write a message and wait for a reply.
- Accept input data (only data entered with a program request command).

FACILITIES OFFERED BY THE CCP

The CCP performs the complex control program services that make the communications-based system easy to use for terminal operators, application programmers, and the system operator.

The CCP performs three types of control program services:

- Communications management
- File management
- Program management

Communications Management

CCP communications management includes all services related to requests from user programs and CCP programs for terminal I/O. The CCP does not actually perform the physical I/O, but performs services for the requester that simplify the use of the MLTA IOCS, MLMP BSCA IOCS, and BSCC IOCS routines that perform the physical I/O. Among the communications management services performed by the CCP are:

- Terminal monitoring and selection.
- Buffer management.
- Symbolic terminal naming.
- Line scheduling.
- Data code translation.
- Terminal testing.
- Other services that allow the application programmer to be largely independent of differences between individual terminals.

Terminal Monitoring and Selection: Terminals that are designated as command terminals during CCP assignment are monitored for commands by CCP communications management. For multipoint lines, the user specifies a polling list during CCP assignment (see index entries: *BSCALINE statement* and *MLTALINE statement*). This list gives the order in which the terminals on a line are to be polled (interrogated) for data and commands when none of the terminals on the line are busy.

For BSCA switched lines, the user can specify a list of valid switched line identification characters. When a connection is established to a terminal on the line, the identification characters of the terminal are validated against the list.

Communications management also monitors program-selected terminals for input and selects them for output. *Selection* is the specific addressing of a terminal by communications management to transmit output data to the terminal.

(See index entry: *terminal types* for additional information about command terminals and program-selected terminals.)

Symbolic Terminal Naming: CCP communications management allows application programs to refer to terminals by six-character symbolic names. This allows the application program to be independent of the specific terminal among a group of like terminals (all are 2740 Model 1; all are 3270; . . .) with which it is communicating. Communications management associates the symbolic name with the physical address of the terminal based on the entries in an internal table. This table is filled in during CCP assignment (see index entry: *TERMNAME statement*). The actual terminal assigned to a symbolic name can be changed by the system operator during the running of the system (see index entry: *assign a symbolic terminal name to a terminal*). To change the name of a command terminal, the terminal operator must also enter a command (see index entry: *name command*).

Communications Service Requests: The CCP provides a subroutine to application programs written in RPG II, COBOL, and FORTRAN IV. These programs call this subroutine whenever they require a communications service. This communications service subroutine puts the user's request into a standard format (independent of the language of the request) that can be interpreted by the CCP. See *Writing Communications Programs.*

After encoding the information for a communications service request, CCP communications management calls the IOCS routine to perform the physical I/O. The CCP schedules I/O operations, determining which request to honor next by chaining the parameter lists provided by the various requesters.

Data Code Translation: CCP communications managemer. translates the transmission line data code to the internal EBCDIC code required for System/3 processing and vice versa. However, the user can specify that the translation should not be made (see index entry: *translation*).

Buffer Management: In addition to the line buffers for each communications line, CCP maintains and manages a common main storage TP (teleprocessing) buffer (hold) area. This aids in carrying out TP operations requested by both user programs and system tasks. CCP communications management reserves and releases portions of this buffer area for input and output data. Because the buffers are allocated on an as-required basis, a program can be temporarily suspended if sufficient main storage is not immediately available to satisfy an I/O request to a terminal. The program is resumed when sufficient main storage becomes available.

The size of this TP buffer area is specified by the user during the assignment run. The user also can change the size of this area during CCP startup.

Online Terminal Testing: With CCP communications management, the terminal operator or system operator can initiate MLTA, BSCA, and BSCC online terminal testing. (See index entry: *start and stop online terminal test.*) Results from the MLTA online test appear at the terminal for the terminal operator to analyze. BSC online test results can appear at the terminal (depending on the terminal type) and are also logged on the system output device or in the system history area for the system operator to analyze.

Note: The system operator can initiate a BSC online test only to another processing unit or to the BSCC attachment (wrap test).

File Management

CCP file management includes all control functions provided by the CCP related to disk and unit record files. CCP file management (DSM with Program Number 5704-SC2) handles the scheduling problems when two or more concurrently-executing programs are trying to access the same physical file. In addition, CCP file management protects data in this file from errors that could result from conflicting operations by contending programs.

Opening and Closing Files: CCP begins its file management function during CCP startup (see index entry: *startup*) when all files available to potential programs are opened. The CCP retains sufficient file information in main storage; therefore, when a user program requests the file, minimum time is required to perform open operations for the program. When the requesting program is finished with the files, they are closed by CCP file management. In reality, they are not finally closed (VTOC updated) until CCP initiates shutdown, the system operator enters a close request using the CCP close program ($CCPCL) or the system operator specifies a KSORT parameter on the PROGRAM statement.

From a batch partition the system operator may close a main data area disk file with utility ($CCPCO). The operator can then perform file maintenance such as sort, delete, enlarge, and reorganize without requiring CCP to shutdown. This file may then be reopened to continue processing in CCP.

Sharing Access to Disk Files: When two or more user tasks are executing concurrently, they can share data files. Sharing of data files is managed by CCP file management. (DSM handles file sharing under Program Number 5704-SC2.) Some tasks can share files while others cannot, depending on how they process the file. The general rules for file sharing are explained in the following chart:

Method of Processing	Type of Sharing Allowed
Ordered indexed load; consecutive output load; consecutive add	*Serial sharing:* the using task must complete before another task can use the file. Ordered indexed loading, if used, must be the first access of the file and can be done only once per run of CCP
Unordered indexed load	*Serial sharing:* file can be shared only by programs doing unordered indexed load.
Indexed sequential add (ISA); indexed sequential update and add (ISUA)	Records can be added, using one of these methods, only once per run of CCP (unless the file is closed and re-opened by the system operator). While the adding program is in progress, sharing is *not* allowed. After the adding program is completed, normal sharing of the file is allowed. Also, no other indexed adds can precede ISA or ISUA (unless the file is closed and re-opened by the system operator). However, if a subsequent program accesses the newly added records, it must be coded as an indexed random add program even though it does not add records.
All other methods	*Concurrent* sharing is allowed.

Managing Physical and Symbolic Files: CCP file management is also responsible for associating the proper physical file with a symbolic file. This is done when the terminal operator enters a file specification command prior to the program request. (See index entry: *file specification command* for a description of symbolic files.)

Program Management

The CCP program management functions include: verifying terminal operator requests for programs, loading programs, allocating system resources to programs, initiating program operation, deallocating system resources, purging programs from the system, maintaining a record of the number of requests for each program, task chaining, and setting the priority of tasks.

Program Requests: When a terminal operator or the system operator requests execution, the CCP verifies that the requested program was defined to the system during assignment (see index entry: *program statement*). If the program was defined, main storage is allocated for the requested program. If sufficient contiguous main storage is not available, the request is rejected unless the requesting terminal is in /Q status. In this case, the program is placed on a wait queue until main storage space becomes available. When the space is available, the request is sent to allocation.

Allocation/Initiation/Termination: When a pending program request reaches allocation, the CCP attempts to allocate system resources to the program based on the description of the program given during the CCP assignment stage. The CCP allocates system resources and initiates program execution when all the required resources (main storage space, disk and unit record devices, and terminals) are available and file usage requirements can be met. If a program requires several resources but some are not available, the resources that are available are not bound to that program until all are available.

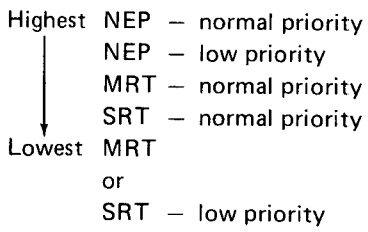
Unit record devices are allocated to only one program at a time if this was specifically requested during assignment (see index entry: *program statement*). The printer can be shared by multiple programs if explicitly permitted by a program statement. Disk files can be shared if the processing required by the programs does not preclude sharing (see index entry: *sharing access to disk files*).

When a program completes processing, the CCP releases the resources used by the program and makes main storage available for another program.

Program-Requests Count: If this option is selected during generation, CCP program management accumulates a record of the number of times each program is requested. (Not all program requests cause the program request count to be incremented. The exceptions include requests for an already active MRT program and certain rejected requests.) However, running the assignment program to update or delete an active assignment set causes program counts to be lost. Also, the count in $CCPFILE is updated during CCP shutdown. The count can be printed and/or reset to 0 by running the assignment list program (see index entry: *assignment list program*).

Task Chaining (Program Number 5704-SC2 only): When a CCP program is ready to request initiation of another CCP program, the requesting program must have a chain task request using the communications service subroutines. The requested task is then handled by the CCP program request function and is executed when all required resources are available.

Task Priority: The priority of user tasks under CCP is handled according to the program type. A never-ending program (NEP) is given the highest priority, a multiple requester program (MRT) is given the next highest priority, and a single requesting program (SRT) receives the lowest priority. Program Number 5704-SC2 provides for selection of a low priority on the Assignment PROGRAM statement. An NEP with this low priority has a priority lower than NEPs of normal priority but higher than MRTs or SRTs. An SRT or MRT with low priority receives a priority lower than SRTs of normal priority. The established order of priority is as follows:



Chapter 5. Designing Your Communications-Based System

This chapter introduces you to the factors you must consider in designing a communications-based information processing system using the System/3 Model 15 and the CCP. For more information, refer to *IBM System/3 Communications Control Program System Design Guide*, GC21-5165. You should regard the CCP and the communications system as a means to an end, the end being increased accuracy and faster flow of information, greater efficiency in the organization, and increased volume of work. During the preinstallation activity, define the overall objectives of the communications system, define the requirements of all departments that will use the system, and produce a detailed plan for preinstallation and installation activity. You should plan applications, use of terminals, data files, programs, and equipment needs before installation of the CCP to speed the installation of the CCP and reduce errors.

Publications referred to in this chapter and elsewhere in this manual should not be considered a complete bibliography for designing a communications-based system. Many publications are available to describe in detail the design factors summarized in this chapter. IBM systems engineers and marketing representatives can help obtain these publications and can also assist in arranging education classes concerning System/3 communications system design. Although many publications currently available are for larger systems and applications (such as airlines reservations systems), the basic types of applications and techniques of data communication also apply to System/3 with the CCP.

APPLICATIONS

The basic element in any system design process is determining what the applications of the system will be. You probably have already determined that you need to process information more accurately and efficiently. For example, perhaps the flow of information and the processing required to perform weekly payroll for a growing number of employees in scattered locations, performing different jobs has become inefficient. An information bottleneck has developed in the central payroll office. A network of terminals communicating the payroll information to the central processor in the payroll office eliminates the bottleneck. To do this, payroll data is sent as soon as it is available, and processed immediately. Payroll inquiries from the remote locations can also be processed immediately.

Assume payroll is identified as an application for your communications-based system. The next step is to determine what related applications can be performed by your system. Perhaps you have a need for more immediate processing of personnel information, which is closely related to the payroll information. Sorts on information in the files may be required. Information in the personnel files maintained by the central processor can also be made available to inquiries from the remote terminals. Perhaps you can use the production totals for individuals in the separate work areas in production accounting.

TERMINALS

When you have identified a major application and related applications, you can determine preliminary locations for terminals. Perhaps you locate separate terminals in a manufacturing area, an assembly area, a warehouse, area, a shipping and receiving area, a sales office, and the central business office. When you have determined the preliminary locations, you can consider the other possible uses for the terminal in each location. In the manufacturing and assembly areas, perhaps you have a need for parts control; in the warehouse area you may have a need for inventory management; in the shipping and receiving area, a need for a shipping order and invoice processing; in the sales office, a need for purchase order and service order processing and sales analysis; in the central office, accounts receivable, billing, and general accounting.

The possibilities for applications in any kind of organization are many. A terminal in one location can serve more than one application.

In choosing terminals for different locations and uses, you should consider the following:

- Is the terminal to be shared by operators with different requirements? If so, the terminal type must be compatible with all requirements.
- Is a heavy workload expected? If so, perhaps more than one terminal is required at the location or a faster line speed is required (line types are described in *IBM General Information—Binary Synchronous Communications,* GA27-3004).
- Will a display-type terminal (IBM 3270) or a typewriter terminal be needed (all terminals on the same multipoint line must be compatible)?
- Will a high volume of activity of the terminal justify special features on terminals, such as the buffer-receive feature on the IBM 2740, Model 2?

Note: Uses for terminals in a communications-based system (data entry, inquiry, inquiry-with-update) are defined and described briefly in the *General Information Manual,* GC21-7578.

DATA FILES

When applications of the communications-based system are determined, plans must be made for the data files to support those applications. The basic decisions to be made initially are:

- What separate files are needed?
- How should the files be organized to best satisfy the different uses to which the files will be put?
- If indexed files are being used, are records to be added during a CCP run? Are the added records to be accessed during the same CCP run?
- Will tasks and batch partitions need to share files?

Many applications require separate files containing current information, and to-date information. In a payroll application, for example, the following files might be required:

- A file of daily information (hours worked, production).
- A file containing information about each employee and the to-date information.

Other ways of differentiating between files could be:

- Separate files for separate branches of an organization, such as schools in a school system.
- Separate files for different products.

After you identify the separate files you need, you must find the best file organization for each file according to its use. For example, files that are normally used for online processing might be subject to batch processing when the files are loaded. Analyzing the percentage of online processing time versus the percentage of batch processing time aids in selecting the file organization that is most efficient overall. For example, if processing is 90% online and 10% batch, your choice of file organization should be weighted toward direct organization. That is, you can devise an efficient method of deriving relative record numbers. (See *IBM System/3 Disk Concepts and Planning Guide,* GC21-7571, for a description of direct files). If processing is 50% online and 50% batch, indexed organization is the best compromise organization. Perhaps you will use the file for online processing in one partition and for batch processing in the other partition. In that case, either direct or indexed organization might apply, since both can be processed either randomly or consecutively.

If you are going to do a sort operation on input files (Program Number 5704-SC2 only), you may want to specify NOSHR in the FILES parameter (of the PROGRAM statement) at assignment time to control the output of the sort. However, by specifying NOSHR you prevent access to the input files while sort is running. Therefore, it is important to plan when to perform your sort operations.

The *IBM System/3 Disk Concepts and Planning Guide,* GC21-7571, contains information that will aid you in choosing a file organization and planning disk files.

PROGRAMS

You must also plan how your applications are to be performed by your communications programs. Programs should be single function if possible. You can debug your program easier and use less main storage. It also provides space for adding new applications.

You should consider the provision of the CCP for physical files and symbolic files in applications that involve processing different files on different runs (see index entry: *symbolic files*). If programs are to access records added to an indexed file during the CCP run, the programs must be written to include the add function (refer to index entry: *program assignment statement*) even when the program will not add any records.

For information on program structure and overlays, see the following publications:

- *IBM System/3 Overlay Linkage Editor Reference Manual,* GC21-7561
- *IBM System/3 Subset American National Standard COBOL Reference Manual,* GC28-6452
- *IBM System/3 RPG II Reference Manual,* SC21-7504
- *IBM System/3 FORTRAN IV Reference Manual,* SC28-6874
- *IBM System/3 Basic Assembler Program Reference Manual,* SC21-7509

When designing a program to run under the CCP, you must consider the program's use of terminals. Should the program service a request from one terminal at a time or from multiple terminals? Should terminals be selected by the program or should individual terminals request the program when they need it? Perhaps there are security considerations that indicate the program should have a single requester or a limited number or requesters. How long will the program remain in main storage? Is it a brief inquiry application or a more time-consuming, data-entry application? If the program will be used frequently, perhaps multiple requesters should be allowed, or perhaps the program should be written as a never-ending program. Program design under the CCP is described in *IBM System/3 Communications Control Program Programmer's Reference Manual,* GC21-7579.

If you run applications under the CCP concurrently, you must plan the program so they run smoothly together, especially during peak times. Programs running together cannot, for example, each require dedicated use of unit record devices. Perhaps you should run batch programs only during particular times of the day. You must plan peak processing times so that system resources are available to the programs that must execute.

You should test individual programs and the entire system in advance to ensure that all applications execute as planned and to evaluate the performance of the system against its planned performance. If the system does not perform as planned, review your program structures, file organizations, and placement of files and programs on disk.

ESTABLISHING THE SYSTEM

The CCP is designed to suit diverse data processing environments involving online applications. Telecommunications line and terminal configurations, and the configurations of central processors that host them, vary greatly among the users of the CCP. Different users have different requirements in functional and performance characteristics of their telecommunications subsystem. The CCP, as distributed, consists of a set of instructions that can be tailored to the needs of the user.

The CCP is established in the following stages:

- Generation
- Assignment
- Operational system startup

Generation of the CCP by the user is the first stage in tailoring the system to his needs. This stage requires the processing of a number of generation control statements, a series of link-edits, and a set of disk utility operations. During this stage a set of the CCP system is created that defines the functions that this version of the CCP can perform.

Once generated, the CCP is still not bound to a specific set of user programs, data files, or terminal assignments. The set of programs, data files, and terminals of any CCP system will probably vary throughout the life of that version of the CCP, even as the required functional abilities remain constant. Therefore, a procedure separate from generation, called *assignment* is provided for the establishment and modification of these definitions.

Like generation, the assignment stage is performed as a normal System/3 operation: its specific function is the creation, modification, or replacement of a set of control tables used by the CCP to manipulate user programs, files, and terminal resources. But unlike the generation process, which tends to be lengthy, the assignment stage is rather brief, requiring only the reading, interpretation, and encoding of straightforward user specifications, and the writing of these to a disk data file.

The assignment stage identifies programs, terminals, disk files, unit record devices, and symbolic terminal names to be used in a particular execution set for the CCP. As with generation, the programs necessary to perform the assignment stage are supplied as part of the distributed CCP modules. The user can run the assignment stage many times in the development of his telecommunications applications. For example, he might have just developed a new application program that he now wishes to incorporate under the CCP. He might also have different assignment sets that can be used for different runs of the execution of the CCP.

After he performs the generation and assignment stages, the user is ready to operate his CCP system. The system operator now loads the CCP, specifying the assignment set the system will execute under. Certain additional elements for only the current run of the CCP can be specified during *operational system startup.*

OPERATING THE SYSTEM

After the CCP is tailored to the user's environment and needs by the generation and assignment stages, it can be put into operation. The CCP operational stage occurs in three parts: startup, operation, and shutdown.

User application programs must be placed in an object library before CCP startup. If a new program is being added, it must be placed in the object library and the CCP assignment build program must be run to update the assignment file ($CCPFILE) before CCP startup.

If you are running under Program Number 5704-SC2, you can specify EXECFIND-YES (in the PROGRAM statement) in the assignment set. With this specified and with cataloging to an active library allowed, a new or updated program can be found during CCP execution without an intervening shutdown.

Also, under Program Number 5704-SC2, the program CCPFMT can be run at CCP execution time to find formats which have been added or updated since startup, and to update the DFFSFDT value (in the PROGRAM assignment statement). For assignment statement considerations, see // PROGRAM statement in the *Assignment Stage* section of this manual.

Startup

In order to initiate the CCP startup routine, the operator enters the following OCL cards from the reader:

```
// LOAD $CCP,unit

// FILE (one file statement is required for each physical
      .    user file accessed during the current CCP run)
      .
      .

// RUN
```

After the CCP startup routine is loaded into main storage, it asks questions of the system operator that allow him to exercise several options. See index entry: *initiating the CCP* for a description of the questions.

Startup performs the following initialization functions:

- Loads the resident CCP.
- Updates various tables from the current assignment set in $CCPFILE.
- Builds control tables in main storage for files and terminals.
- Verifies that adequate main storage is available.
- Allocates and opens the files that will be used during the current run.
- Locates all user programs that might be requested during the current run (unless EXECFIND-YES is specified in the PROGRAM statement at the assignment time).
- Finds sort modules if they are required.
- Loads resident modules as required.
- Opens communications adapters and lines.

Startup issues diagnostic messages if the system requirements for startup are not met, if the user entered invalid instructions, or if it cannot complete its initialization for some other reason.

Operation

During its operation, the CCP manages the environment in which telecommunications application programs run and provides services upon which they can call. The management functions of the CCP are of three types: communications management, file management, and program management.

Shutdown

CCP shutdown is initiated by command from the system operator (see index entry: *shutdown command*). After the shutdown command, the CCP goes to EOJ when programs running under CCP finish executing.

The following specific operations are performed at CCP shutdown:

- Executing programs are notified to go to end of job.
- Communication lines and adapters are closed.
- Disk data files are closed.
- If the program request count option was selected during generation, the number of requests for each user program during the CCP run are added to the previous accumulated count. For Program Number 5704-SC2, if the set has been changed since startup, the addition will not be done.
- The CCP sends a closing message to the system operator.
- The CCP exits to the system end-of-job routine.

UPDATING THE SYSTEM

In time, the uses of the communications system might change. You must plan for possible updating and additional tailoring of the system as you gain experience with the system. You should make allowance, for example, for the turnpike effect, a phenomenon observed after the first modern superhighways were planned and built. Use of the new highways was greater than anticipated, because drivers tended to stop using the old routes in favor of the new highway. Overall traffic flow increased beyond expectations because of the convenient new highway. The turnpike effect has been observed in previous communications systems and is a factor to consider in planning for a System/3 communication-based system.

TOTAL EQUIPMENT NEEDS

When you have considered all factors—applications, use of terminals, data files, programs, and provisions for system updating—you can make decisions concerning the total equipment needs of your organization:

- What are the total storage requirements of the supervisor, the CCP, the TP buffer, and the user area?
- How much disk storage is needed and of what type? How much space is needed for libraries, how much for files? (For detailed storage estimates for the CCP, see Appendix F, *Storage Estimates.*)
- What terminals are needed?
- What communications equipment and lines are needed?
- What is the total cost?
- When can deliveries be made?
- What unit record devices are needed and how fast should they be?
- Are you spooling the unit record device input or output?

Information concerning teleprocessing equipment characteristics communications concepts, common carriers, network design, and other useful information is contained in the following publications:

- *IBM Data Communications Primer,* C20-1668
- *IBM System/360 Introduction to Teleprocessing,* C30-2007

DISK SYSTEM MANAGEMENT (DSM) CONSIDERATIONS

The CCP operates in conjunction with DSM and uses DSM facilities whenever possible, including the I/O supervisor for disk and unit record I/O devices.

Certain constraints are placed on the existing DSM programming support.

1. CCP user programs use additional DSM task control blocks in the DSM nucleus, which must be requested via CCPUT keyword of DSM generation.
2. Programs not link-edited for CCP cannot run in the CCP partition.

3. For program partitions:

   - CCP can be used in only one partition at a time. However, when CCP is running in one partition, user programs not controlled by CCP can run concurrently in the other partition(s).

   - CCP should run in the highest priority partition to maximize throughput.

   - A program running in the non-CCP partition can use a teleprocessing adapter (MLTA or BSCA) concurrently with the execution of CCP. However, this adapter must not be supported by the same assignment set currently used by CCP. MLTA and BSCC are mutually exclusive features.

4. Multivolume disk data files are not supported by Program Number 5704-SC1. Random access to multivolume files is supported by Program Number 5704-SC2.

5. To reduce degradation from disk access contention, consider including more than one disk drive in the CCP configuration of the system.

6. Spooling while running CCP allows multiple CCP user programs to use spooled devices. Also, programs are not limited by the device speeds. (See *Spooling Considerations* later in this chapter.)

7. For disk files:

   - All disk files to be processed during a CCP run must be online, and specified in OCL file statements following the OCL load statement to load CCP.

   - All disk files are normally opened at CCP startup and closed at CCP shutdown. Because of this, no indexed-file, key-sort occurs until CCP shutdown. Programs wishing to access all records added to an indexed file should include indexed *add* data management in their programs. However, files can be closed and a key sort performed via the system operator file closing facility ($CCPCL). A file may also be closed through the use of the KSORT parameter on the PROGRAM statement, assuming no other program is using the file at end of job. Records added prior to the close request are merged with existing records and available to all programs when the file is reopened ($CCPOP).

   - (Program Number 5704-SC2 only.) Main data area disk files may also be closed for file maintenance from a batch partition using a system operator utility ($CCPCO). $CCPCO executes in a non-CCP partition. After $CCPCO has been successfully executed, the operator can use other programs to enlarge or sort the file and delete records from the file. $CCPCO can then be used to reopen the file for CCP access.

   - CCP permits concurrently running programs to update the same file or to do indexed add to the file. To do this, CCP (5704-SC1) or the system (5704-SC2) protects the block of sectors containing the record until the program releases the block of sectors. The block is released when a never-ending program, multiple requesting terminal, writes the block to the disk, reads another block, goes to end of job, or issues an accept input operation. However, a possible lock-out condition can occur if the program never releases the block.

8. A unit record device cannot be shared by concurrent programs running under CCP. (*Exception:* the printer can be shared by concurrent programs running under CCP if specified by the assignment control statement.) When a program ends, the device is available to another program.

9. Checkpoint/restart is not supported under CCP, but can be used in the non-CCP partition.

10. Inquiry (rollout/rollin) is not supported under CCP but can be used in the non-CCP partition (for Program Number 5704-SC1 only).

11. For user programs:

    - User programs must be defined to CCP within an assignment set.

    - Teleprocessing must be done through CCP.

    - A terminal requesting a program is tied to that program until released by the program.

    - User programs with names beginning with $ are not permitted to run under CCP except $CCPDD, $CCPCL and $CCPOP which are noncancel programs.

    - To use a console, you must use CONSOL as the terminal name.

- Programs cannot be larger than 32K including DFF PAS; for Program Number 5704-SC2, the 32K does not include external buffers or any area for memory resident overlays. (32K is the maximum size because each program can only address 64K; each must address 16K of supervisor, and 16K of CCP leaving 32K for the user.)
- The program size is rounded to a multiple of 2K with a minimum of 4K.
- CCP does not support tape units.
- CCP does not support user programs accessing the 3284 attached to the processing unit.
- User programs must be link-edited at 32K (X'8000'). This allows faster program load and reduces the load on the DSM transient area.
- For Program Number 5704-SC2, disk DTFs (define the files), IOBs (input/output blocks), and internal buffers cannot be located at an address greater than logical X'DFFF'.

12. The assembler user cannot:
    - Use a special allocate.
    - Do a read using system input.
    - Write to the scheduler work area (SWA).
    - Write to the volume table of contents (VTOC).
    - Execute privileged instructions.

Spooling Considerations (Program Number 5704-SC1)

For printing: If the printer is allocated to only one subtask and a // PRINTER control statement with a DEFER-NO parameter was specified at CCP startup, the spooling routines close the spool print file when the CCP CLOSE for a subtask is called.

For punching: If DEFER-NO is specified on the // PUNCH control statement, each subtask which closes a punch device causes the spooling routines to close the spool punch file at the same time.

The jobname and stepname on the print and punch spool queues are the same as the partition jobname and stepname.

Spooling Considerations (Program Number 5704-SC2)

For reading: Only a nonspooled partition can read data directly from the card device that is the spool reader. To read data, the spool reader must be terminated and data must be ready in the card input device.

A nonspooled partition can indirectly read data from the spool reader. If the card input device for a spooled partition is not the spool reader, requests for data from the spool reader by a program generate an 'SP UT NR' message (input device is spool reader). If a zero option is selected, the spooling routines use as input data the job stream of the first job on the reader queue scheduled for execution in the partition requesting data from the spool reader. If the data includes a job card, the job card will be used as the first card of data.

If the card input device for a partition is the spool reader, then the data for a program must be put behind the // RUN statement of the program using the data, as shown in the following example.

```
// SAMPLE JOB
// CALL PROG,F1
// RUN
  data
  .
  .
  .
/* (end of data)
/. (end of job)
```

For printing: If the printer is allocated to only one CCP task, the spooling routines close the spool print file at the completion of that task, unless CLOSE-NO is specified on the PRINTER OCL statement at or before CCP startup.

For punching: Each task that closes a punch device causes the spooling routines to close the spool punch file at the same time unless CLOSE-NO was specified on the PUNCH OCL statement before or during startup.

When the output on a spool print or punch queue is produced by a program running under CCP, the terminal name is used as the jobname, and the stepname is generated by taking the program name and appending a sequence number. The sequence number starts at 01 and is increased by one for each jobstep placed on the queue. After 99 is used, the sequence number is reset to 00. A separate counter is used for the print and punch queues.

If a CLOSE-NO parameter is specified on a PRINTER OCL statement at or before CCP startup, the partition jobname and stepname are used to identify tasks on the spool queue.

If the 1403 is the system log device for the CCP partition, the partition jobname and stepname are used to identify CCP tasks on the print queue; after a CCP task causes the spooling routines to close the printer, terminal name and program name are used to identify a CCP step on the queue.

If the printer is allocated to more than one task at a time, the identification on the print queue will be that of the first task to put data on the queue.

CCP Trace

Trace TYPE-CCP is an optional feature of the DSM system trace, which provides a log of system activity. This trace can be useful in determining the cause of problems. The DSM interrupt trace is a separately loaded program that must be executed before CCP startup. Trace TYPE-CCP is invoked by specifying CCP on the TYPE parameter of the trace control statement. For Program Number 5704-SC2 only, and OCC loadable CCP trace function may be initiated after CCP startup and before CCP shutdown. This CCP trace facility provides trace entries formatted like those created by the DSM trace ($TRACE) with TYPE-CCP and TYPE-ALL. For further information on the CCP trace facility, see the *IBM System/3 Model 15 Communications Control Program Data Areas and Diagnostic Aids*, SY21-0040. For a detailed description of the operation and trace table formats, see the appropriate data areas and diagnostic aids or program logic manual listed under *Related Publications* in the *Preface*.

MULTIPLE PARTITION CONSIDERATIONS

The CCP is partially inhibited in its processing when the non-CCP partition(s) is using the DSM system transient area and the CCP partition also requires the transient area. This would most commonly happen when the system input transient (SYSIN) is in the transient area for the other partition (for example, for job initiation). CCP is also partially inhibited when the screen is used for the display command and the display is not immediately canceled or when PF10 (OCC) is keyed and there is a delay in entering data.

The DSM transient area is also used when terminating a CCP user program.

Sharing Files Between Partitions

For Program Number 5704-SC1, CCP update and add files can be accessed and updated by the non-CCP partition if no CCP programs are currently updating or adding to the same file. However, records added to a file by CCP programs cannot be accessed by the other partition until shutdown, or until the file is closed by the system operator (see the index entry: *Disk File Close/Open Facility*).

For Program Number 5704-SC2, enhanced file sharing is available.

3270 DISPLAY FORMAT FACILITY (DFF) CONSIDERATIONS

The 3270 display format facility (DFF) is a facility of the CCP that is selected separately during the CCP generation. The facility allows programs written in RPG II, COBOL, FORTRAN IV, and Basic Assembler to control the display or printer format for the 3270 information display system. The DFF makes it possible to control the display or printer format and perform operations involving data fields in the format directly from the application program, similar to performing the operation with any other terminal supported under the CCP.

The DFF is composed of: the display format generator routine $CCPDF (DFGR for single or multiple format builds), the printer format generator routine $CCPPF (PFGR for single or multiple format builds), the display format test routine $CCPDT (DFTR), and the display format control routine (DFCR). The DFGR or PFGR, which is executed prior to the CCP startup, processes special DFF specifications, builds display or printer formats, and stores the formats in an object library. The DFTR is a stand-alone program executed prior to CCP startup which displays DFF formats on a 3270 Information Display System for testing the screen or printer layout. DFTR cannot be used if BSCC is the only communications interface in the system. The DFCR processes requests for DFF services issued by application programs running under the CCP.

The DFGR or PFGR operates in either program partition but not concurrently in both program partitions. The main storage required for execution is always 18K. The DFGR or PFGR cannot place new formats in a library if the other partition is using temporary entries, or doing library functions on the same pack. If DFGR or PFGR is running with CCP executing in the other partition, the format being processed by DFGR or PFGR cannot be placed on the pack CCP was loaded from. If attempted, an EO F0 PP halt occurs.

For Program Number 5704-SC2, the SCP SYSGEN prompt for catalog protection determines whether the format being processed by DFGR or PFGR can be placed on the pack CCP was loaded from.

For additional discussion of the DFGR and DFCR, see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579.

Components within the 3270 system that are supported are as follows:

Component	Function	Model Numbers
IBM 3271	Control unit	Model 1 and Model 2
IBM 3274	Control unit	Model 1C[1]
IBM 3275	Display station	Model 1 and Model 2
IBM 3276	Control unit display station	Model 2[1]
IBM 3277	Display station	Model 1 and Model 2
IBM 3278	Display station	Model 2[1]
IBM 3284	Printer	Model 1 and Model 2
IBM 3284	Printer	Model 3[2]
IBM 3286	Printer	Model 1 and Model 2
IBM 3287	Printer	Model 1 and Model 2
IBM 3288	Line printer	Model 2
IBM 3289	Line printer	Model 1 and Model 2[1]

Special features of the 3270 system supported are: the selector pen, the audible alarm, and the operator identification card reader.

The 3270 is supported as a remote attachment to the System/3 Model 15. Communications between the System/3 and the 3270 are maintained using the binary synchronous communications multipoint data link mode of operation. All operations that can be performed with the 3270 in the remote operation are supported except for the read-type and general poll commands. Polling sequences are used for remote read operations.

[1] In 3270 compatability mode only.
[2] 3284 Model 3 attaches to 3275 display station only.

Using the Same Terminal with DFF and Non-DFF Programs

A terminal attributes set (defined at assignment time with the TERMATTR assignment control statement) is in effect when a terminal is communicating with user's program under the CCP. There are a number of ways of controlling the attributes set of a terminal:

1. The first TERMATTR statement assigned to a terminal in an assignment set is the *default* attributes set of a terminal. When neither of the following two methods are used, this is the attributes set in effect for the terminal.

2. The attributes set of a required terminal can be specified for the terminal during its attachment to the program via the program assignment control statement, TERMS parameter. When the terminal is no longer attached to the program because of an end-of-job condition or a release terminal operation (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579), the default attributes set is in effect.

3. The attributes set can be specified for the terminal during its attachment to the program via the Acquire Terminal operation (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579) with the set terminal attribute modifier. A terminal already attached to a program with one attributes set can be re-acquired to change the attributes set. The default attributes set is in effect when the terminal is no longer attached to the program.

TERMINAL SECURITY CONSIDERATIONS

If you have no terminal security feature built into your system, each command terminal can issue program requests and other commands to the system after the sign-on command has been entered at that terminal.

If you are concerned with security of access to the system, you can include a password feature in the CCP at generation (see index entries: *$ESEC—terminal sign-on security* and *system assignment control statement*).

In a system with password protection, no requests are accepted from a terminal until its operator presents the current password and the CCP verifies it. Once an operator signs on with the password, he can make any number of requests without repeating it. If the operator is not always at his terminal, he can sign off the terminal anytime he leaves it. Once he has signed off, the CCP requires any further use of the terminal to be accompanied by the password. Thus if other, possibly unauthorized, persons gain access to that terminal, they cannot gain access to the system.

The valid password for the current run is established in an assignment set, but can be changed by the system operator at startup of the CCP (see *Startup Procedure* in the *IBM System/3 Model 15 Communications Control Program System Operator's Guide*, GC21-7619). The password can change at startup on every run, or can be retained for days or even weeks. In any case, only terminal operators who know the password are permitted access to the system.

If the user wishes to write his own terminal sign-on security routines rather than use the CCP password facility, there are stringent requirements his routines must observe regarding the interface with the CCP (see index entry: *$ESEC —terminal sign-on security*).

All sign-on attempts are logged on the console.

Chapter 6. Generation Stage

CCP generation is the process whereby the user selects the portions of the distributed CCP that provides the capabilities he wants in his version of the CCP. Generation is the first stage in creating the CCP, when the user establishes its maximum capability. Further selection is done at assignment time and at operational startup.

The user describes his system configuration and the functions he wants by a series of statements consisting of keywords with associated values. Some of these statements describe the system configuration (main storage size, terminal and line capability) within which the CCP operates. Other generation statements specify the capabilities of the CCP and whether the CCP is to have certain optional features such as password sign-on and user program-request counts.

Generation creates and initializes an assignment file ($CCPFILE), whose contents—the specifications of an actual terminal configuration, disk files to access, and programs to use—are filled in at assignment time.

PROCEDURE FOR GENERATION

The key step for the user in the generation procedure is to describe the kind of CCP required by modifying the sample generation statements. In order to modify the statements, the user must know certain facts about the system and about the capabilities of the CCP he or she wants to generate. These facts are determined during the CCP system design. Design considerations are given in Chapter 5, *Designing Your Communication-Based System.* The following is a checklist of the facts a user must know before performing generation:

During generation, the user chooses the features for his system. In doing this, the user should carefully check each generation option and specification to see if the system requires it. As a reminder, most features increase the size of the control program which, in turn, lessens storage space for application programs to execute (see Appendix F. *Storage Estimates*).

Generation Checklist

_____ Whether or not the data mode escape feature is to be used, and if so the six user-specified data mode escape characters

_____ Is a program request count to be kept

_____ Is display format facility (DFF) to be supported

_____ Is program request under format (PRUF) to be supported

_____ Is resident accept input to be supported

_____ Is resident open/close to be supported

_____ Is resident program request to be supported (5704-SC2 only)

_____ Is DFF moveout to be implemented (5704-SC2 only)

_____ Is Terminal Name Table (TNT) moveout to be implemented (5704-SC2 only)

_____ Is 328X printer busy facility to be supported (5704-SC2 only)

_____ Are information messages (S type) to be sent to CPUs

_____ Programming language(s) to be supported by CCP

_____ Which 5444/3340 Disk Storage Drive(s) you will use to mount the packs during generation that will later be used for preparing programs to be run by CCP

_____ What type of sign-on security will be used, if any

_____ Length of your security comparison information, if you use your own sign-on security checking routine

_____ Anticipated number of assignment sets to be placed into $CCPFILE

_____ Maximum number of programs and files in an assignment set

_____ Maximum number of terminals planned

_____ Space to be reserved in $CCPFILE ($CCPDUMP for 5704-SC2) for dynamic main storage dumps

_____ Main storage size of processing unit

_____ Disk unit and pack name on which the pack to contain $CCPFILE will be mounted during generation

_____ Disk unit and pack name on which the pack to contain $CCPDUMP will be mounted during generation (5704-SC2 only)

_____ Number of tracks in the CCP dump file ($CCPDUMP) to be used for writing the trace tables if CCP trace is active (5704-SC2 only)

_____ Beginning track location for $CCPFILE

_____ Number of MLTA lines to be supported

_____ Whether or not MLTA input and output will always be translated to and from EBCDIC

_____ MLTA terminal devices to be supported

_____ MLTA line transmission codes

_____ BSC lines, line features, and BSC control logic to be included in CCP support

_____ BSC line transmission codes

_____ BSC terminal devices to be supported

_____ BSC interval polling

_____ Is the display adapter to be supported

_____ Is serial I/O channel to be supported (5704-SC2 only)

_____ Disk unit on which disk system management resides (F1 or R1)

_____ Disk unit onto which CCP will be generated

_____ Disk unit(s) and pack name(s) where work file space can be found during generation

_____ Disk unit on which the distribution CCP Modules reside

Following are two methods for doing the CCP generation. The first method uses cards as the primary input medium. The second method is cardless-oriented. This method uses the source and procedure libraries as the primary input medium.

The basic procedure for the card-oriented CCP generation is:

1. A sample control statement deck is punched from the source library of the distribution pack.

2. The sample deck is modified by the user to his specifications and entered as input to the next step of generation.

3. A full job stream to accomplish the necessary functions is punched for the user.

4. The job stream is used to generate your version of the CCP, and the CCP assignment file ($CCPFILE) is ready for the user's initial assignment run.

The basic procedure for the cardless-oriented CCP generation is:

1. Source and procedure members are printed from the distribution pack.

2. The user modifies the sample procedures using the $MAINT modify function for the system configuration as input to the next step of generation.

3. The user enters the CCP specifications to create source and procedure members used to generate your version of the CCP and the CCP assignment file ($CCPFILE).

The function of the generation stage of the CCP, regardless of the method used, is to:

- Generate modules that require modifying at the source level.

- Link-edit the generated modules and certain other relocatable modules. This is done to create three load modules: two are the resident control program during CCP operations, the other initializes $CCPFILE.

- Copy these and other load modules to the designated CCP production pack. The production pack is the pack from which the CCP is loaded for execution. The CCP production pack can be any pack other than the distribution pack. It might be a DSM system pack and might also be the current system pack during the CCP generation.

- Copy additional selected relocatable modules to the program preparation packs (packs that will be used for compilations and linkage edits of user written application programs that will execute under the CCP).

- Allocate and initialize (but not fully enter information into) the CCP assignment file ($CCPFILE) on the designated pack.

The generation stage assumes:

- That the DSM is properly generated on the system pack, including the appropriate MLTA, BSCA, and/or BSCC I/O macros and subroutines.

  MLTA: The MLTA microcode deck (obtained from the service representative) must be loaded into the object library under the name $MLMC1 on the system pack.

  The BSCC Microcode Deck (obtained from the service representative) must be loaded into the object library under the name $$BSYD on the system pack.

  The MLTA error statistics file (MLTERFIL) must be created and initialized on the system pack. To initialize MLTERFIL, the MLTA feature provides module $$MLFI in the object library. The OCL statements required to initialize MLTERFIL are:

```
// LOAD $$MLFI,XX
// FILE NAME-MLTERFIL,UNIT-F1,
//      RETAIN-P,RECORDS-9,
//      PACK-PPPPPP,LOCATION-nnn
// RUN
```

  BSC: If BSCA or BSCC lines are used, a file on the system pack for logging control station terminal statistics must be provided. To initialize MLTERFIL, the MLMP (multiline/multipoint) feature provides module $$BSFI in the object library. The OCL statements required to initialize MLTERFIL are:

```
// LOAD $$BSFI,XX
// FILE NAME-MLTERFIL,UNIT-F1,
//      PACK-PPPPPP,TRACKS-n[1],
//      LOCATION-nnn,RETAIN-p
// RUN
```

[1] One track is normally specified, but for BSCC, two tracks must be specified.

Note: MLTERFIL need be initialized only once to accommodate BSCC, BSCA, and MLTA statistics. Part of MLTERFIL comprises the BSCC/BSCA terminal log area and is used for logging the control station terminal statistics. Another part of MLTERFIL is used for logging MLTA statistics if MLTA is present. Do not initialize the file twice if you use both BSC and MLTA.

- That during generation the copy of the macro processor ($MPXDV and all its subsequent load modules) and the overlay linkage editor that exists on the CCP pack are used during the CCP generation.

- That an appropriate sized object library and source library were allocated on the production pack (5444/3340 only). See index entry: *disk storage estimates for the CCP.*

- That the object library on the production pack was reorganized and no modules were deleted since reorganization. All the CCP modules should be contiguous on the production pack after the CCP generation. (The CCP has no control over where DSM can place modules if modules were deleted.)

- That if SIOC is to be supported by CCP, the appropriate SIOC I/O modules must be on the system pack. Refer to *IBM System/3 Model 15D Channel Connected Systems Program Reference and Logic Manual*, GC21-5199.

- That $CCP is not running during generation.

Notes:

1. The printed output resulting from generation must be saved in case of required maintenance by IBM Field Engineering personnel. This paper is the only documentation of the user's unique system and the precise sequence of events during this particular CCP generation.
2. The user should consider backup procedures (of his own design) in case the CCP distribution pack or a generated pack are inadvertently destroyed.
3. Generation can be accomplished on a System/3 Model 15 other than the one using the CCP in teleprocessing.

Operational Procedures For Generation

The CCP is distributed on a pack-separate from the distribution of the other components of the System/3 Disk System Management. If you are generating both the basic DSM and the CCP, you must generate the basic DSM first, following the procedures described in *IBM System/3 Model 15 System Generation Reference Manual*, GC21-7616.

To generate the CCP, you must mount the CCP distribution pack on a unit separate from the system pack. You must have performed an IPL (initial program load) from the system pack at some previous point. Unlike the generation of DSM, you can direct the output from the CCP generation to more than one pack. The output consists of:

- Those modules required for the assignment stage and for executing the operational CCP. These modules are directed to the CCP production pack.

- The subroutines to be used on compiling and link editing application programs to be run under the CCP (macros in the case of Basic Assembler language programming). These subroutines are directed to one or more program preparation packs.

Each of the above might be a different pack, or could be the same pack. You specify the disk unit on which each of these packs is mounted.

The CCP generation procedure is dependent upon the presence of the macros and subroutines for BSCA, BSCC, and/or MLTA (as appropriate for the terminal devices to be supported) on the IPL pack.

Figure 3 outlines the procedure for generating the CCP. The procedure assumes that disk system management was generated and disk system IPL was performed previously.

At this point the user must decide whether to use card-oriented CCP generation or cardless-oriented CCP generation. Six steps for the card-oriented generation are:

Step 1 (User, Card Oriented):

The user enters from the system input device (the console or the card device) statements of the following form:

```
// &
// LOAD $MAINT,dsunit
// RUN
// COPY FROM-diunit,TO-PRTPCH,
//      LIBRARY-S,NAME-$CGSMP
// END
```

Note: **dsunit** on the // LOAD statement is the unit on which the DSM system pack resides. The **diunit** in the // COPY statement is the unit on which the CCP distribution pack is mounted.

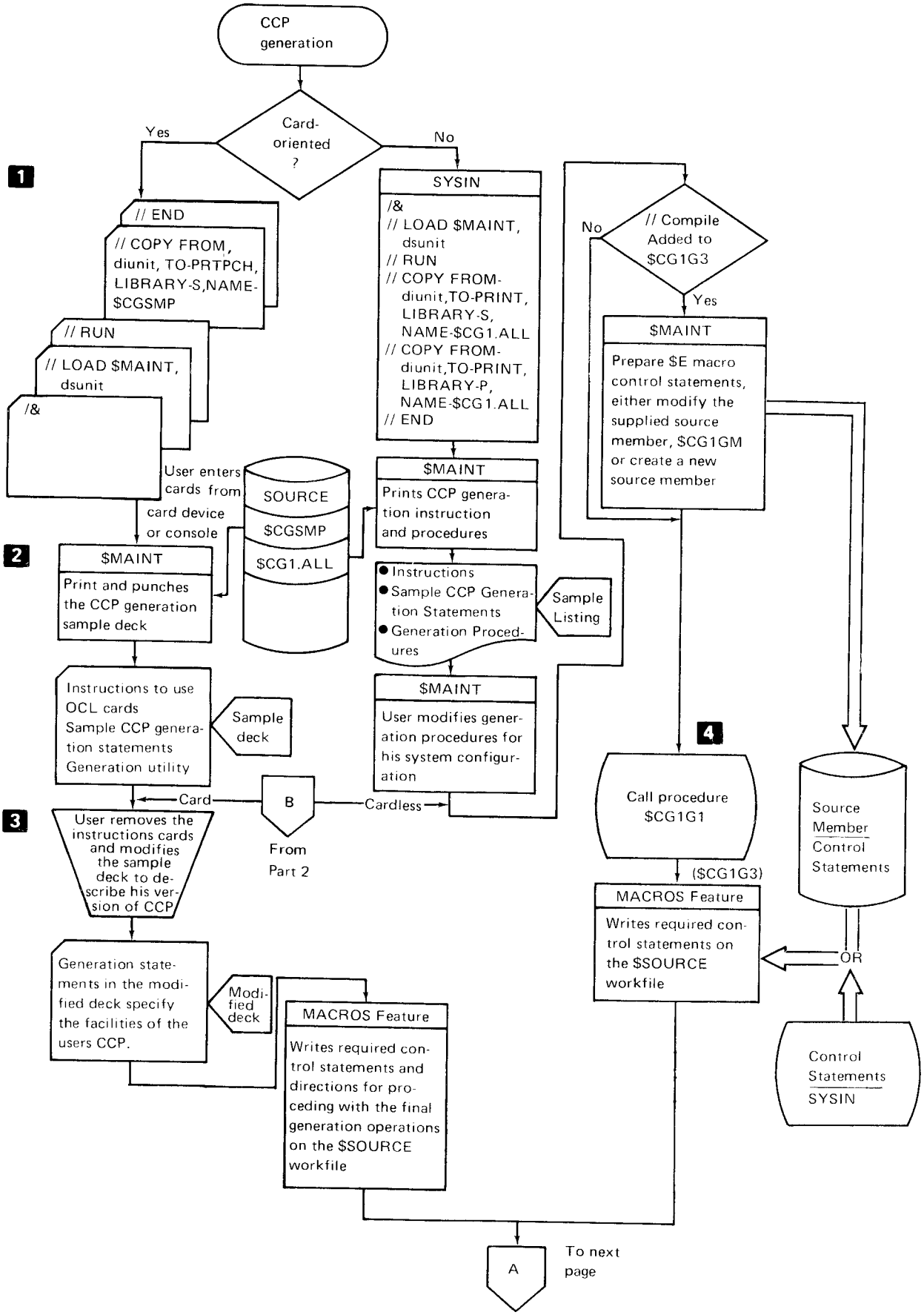


Figure 3 (Part 1 of 3). CCP Generation Procedure (Card and Cardless)

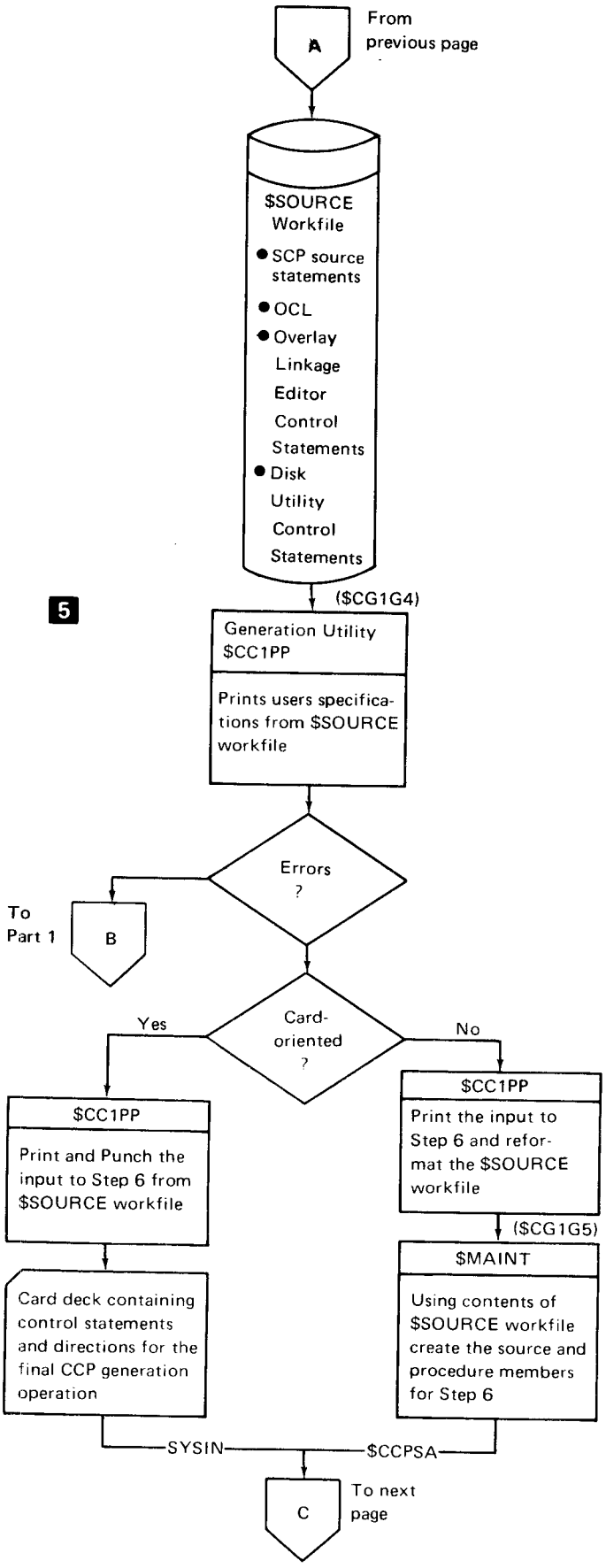


Figure 3 (Part 2 of 3). CCP Generation Procedure (Card and Cardless)

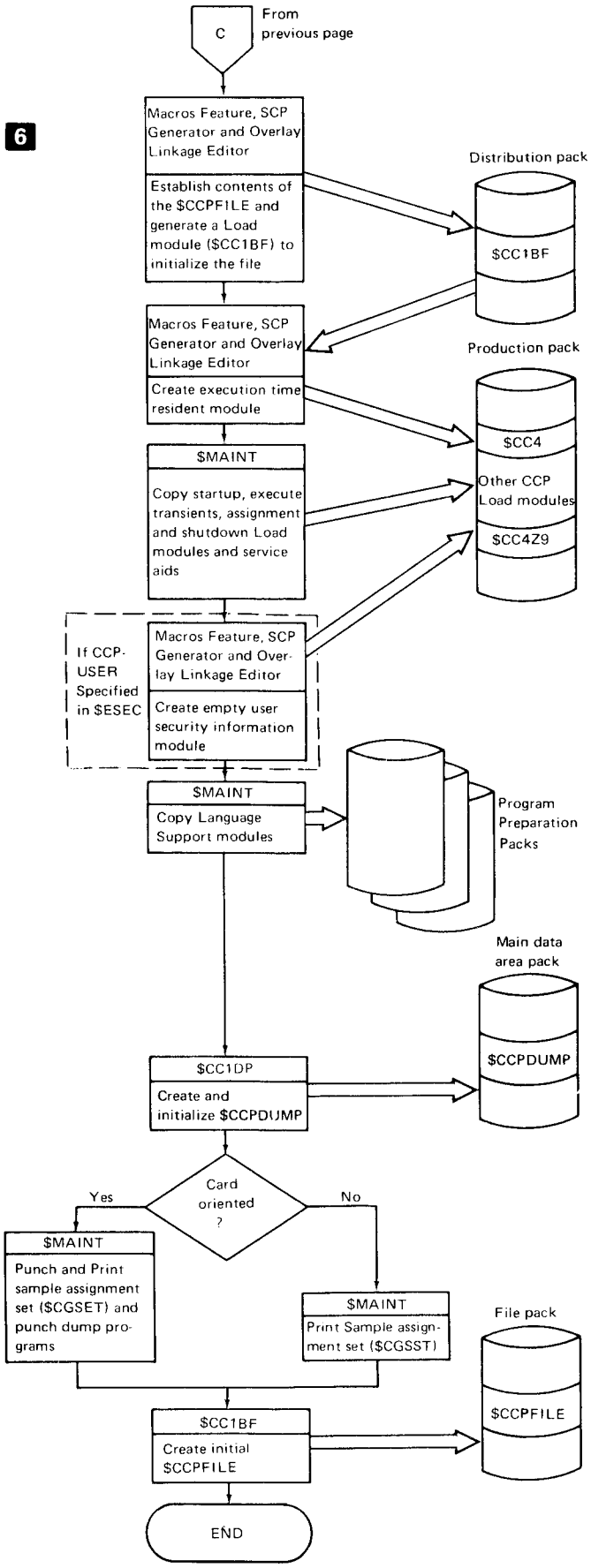


Figure 3 (Part 3 of 3). CCP Generation Procedure (Card and Cardless)

Step 2 (System, Card-Oriented):

The system retrieves $CGSMP (CCP sample generation deck) from the source library on the CCP distribution pack, and then prints and punches the entire member.

The sections to the information printed and punched are:

1. Instructions to the user for modifying the punched deck.

2. OCL to call the macro processor, followed by sample $E generation control statements to generate a CCP system. One card is provided for each operand of a generation control statement. The card generally contains the default value, and provides a comment indicating alternative values for the operand.

3. OCL to call the CCP generation utility program ($CC1PP).

For Program Number 5704-SC2, all // FILE cards punched during generation will specify SHARE-NO. An example of the printout that accompanies the sample generation deck is:

```
$CGSMP
*************************************************************************
*              I N P U T   T O   C C P   G E N E R A T I O N            *
*************************************************************************
*                                                                       *
*                                                                       *
* THESE CARDS -- AFTER YOU HAVE MODIFIED THEM TO YOUR SPECIFICATIONS --  *
*   WILL BE THE INPUT TO THE FIRST PASS OF CCP GENERATION               *
*                                                                       *
*                                                                       *
* DISCARD THE LEADING CARDS OF THIS DECK, FROM THE // COPY CARD AT THE  *
*   BEGINNING THROUGH THE CARD THAT MARKS *** END OF INSTRUCTIONS ***   *
*                                                                       *
* DISCARD ALSO THE // CEND CARD -- THE LAST CARD OF THE DECK            *
*                                                                       *
*                                                                       *
* YOU MUST MAKE THREE KINDS OF MODIFICATIONS TO THE REMAINING CARDS --  *
*                                                                       *
*   1. IN THOSE CARDS THAT ARE MARKED  ++  IN COLUMNS 74-75, REPLACE    *
*      ANY  $$  OR  ##  IN THE CARD WITH THE IDENTIFICATION OF A DISK   *
*      UNIT --                                                          *
*                                                                       *
*      --  $$  WITH THE UNIT ON WHICH YOUR SYSTEM PACK IS LOCATED       *
*                                                                       *
*      --  ##  WITH THE UNIT ON WHICH THE CCP DISTRIBUTION PACK IS      *
*              LOCATED                                                  *
*                                                                       *
*   2. REPLACE CARD NUMBER 00299 WITH A CARD PUNCHED  /*  IN COLUMNS 1-2 *
*                                                                       *
*   3. MODIFY THE GENERATION CONTROL STATEMENTS -- THOSE CARDS          *
*      FOLLOWING CARD NUMBER 00206 -- TO SPECIFY THE REQUIREMENTS OF    *
*      THE CCP YOU WISH TO GENERATE                                     *
*                                                                       *
* IF YOU DO NOT WISH TO USE THE DISTRIBUTION PACK FOR THE REQUIRED      *
*   $SOURCE FILE, SPECIFY THE LOCATION OF THAT FILE BY CHANGING THE UNIT *
*   AND PACK PARAMETERS IN CARDS 00204 AND 00306                        *
*                                                                       *
*                                                                       *
* WHEN YOU HAVE MADE THESE CHANGES, PLACE THE MODIFIED DECK IN THE      *
*   HOPPER OF THE SYSTEM INPUT DEVICE AND BEGIN PASS 1 OF CCP GENERATION *
*                                                                       *
****** END OF INSTRUCTIONS ******************** END OF INSTRUCTIONS ******
*                                                                        00100
*                                                                        00200
***  PROCESS SPECIFICATIONS FOR THE CCP TO BE GENERATED                  00201
*                                                                        00202
// LOAD  $MPXDV,##                                                      ++00203
// FILE  NAME-$SOURCE,RETAIN-T,UNIT-##,PACK-PID001,TRACKS-20,           ++00204
//       LOCATION-386                                                    00205
```

```
// RUN                                                                                    00206
        $EFAC ESCAPE-NO,          -- 'CCCCCC' / X'XXXXXXXXXXXX' --          X    FAC00
              PGMCNT-NO,          -- YES --                                 X    FAC01
              PRUF-NO,            -- YES --                                 X    FAC02
              FORMAT-NO,          -- YES --                                 X    FAC03
              ACCEPT-NO,          -- YES --                                 X    FAC04
              RESOPN-NO,          -- YES --                                 X    FAC05
              BSYPRT-NO,          -- YES --                                 X    FAC06
              MOVTNT-NO,          -- YES --                                 X    FAC07
              MOVDFF-NO,          -- YES --                                 X    FAC08
              RESREQ-NO,          -- YES --                                 X    FAC09
              SIOC-NO,            -- YES --                                 X    FAC10
              OPPRUF-YES,         -- NO --                                  X    FAC12
              CPUMSG-NO,          -- YES --                                 X    FAC13
              LOWCAS-NO,          -- PF1 - PF12 --                          X    FAC14
              TTASK-NO            -- YES --                                      FAC15
        $EPLG LANG-      ,        -- COBOL / RPGII / FORTRAN / ASSEM --     X    PLG00
              PPUNIT-             -- R1 / F1 / R2 / F2 --                        PLG01
        $ESEC SECURE-NO,          -- CCP / USER --                          X    SEC00
              LUSI-0              -- 1 - 4096 IF SECURE-USER --                  SEC01
        $EFIL SETS-1,             -- 2 - 25 --                              X    FIL00
              PROGS-10,           -- 1 - 999 --                             X    FIL01
              DFILES-5,           -- 1 - 192 --                             X    FIL02
              TERMS-1,            -- 2 - 254 --                             X    FIL03
              DUMPS-1,            -- 2 - 99 --                              X    FIL04
              CORE-96K,           -- 128K/160K/192K/224K/256K/384K/512K     X    FIL05
              FLUNIT-   ,         -- R1 / F1 / R2 / F2 --                   X    FIL06
              TRKLOC-     ,       -- VALID TRACK NUMBER / OMITTED --        X    FIL07
              FLPACK-       ,     -- NAME OF PACK --                        X    FIL08
              DPTRAC-6            -- 0 - 1000 --                                 FIL09
        $EMLA LINES- ,            -- 0 - 8 --                               X    MLA00
              XLATE-YES           -- NO --                                       MLA01
        $EMLD TYPE-       ,       -- SEE SYSTEM REFERENCE MANUAL --         X    MLD00
              XMCODE-             -- SEE SYSTEM REFERENCE MANUAL --              MLD01
        $EBSC BSCA- ,             -- 0 - 2 --                               X    BSC00
              DIAL-NO,            -- YES --                                 X    BSC01
              PP-NO,              -- YES --                                 X    BSC02
              MP-NO,              -- YES --                                 X    BSC03
              CS-NO,              -- YES --                                 X    BSC04
              GETMSG-NO,          -- YES --                                 X    BSC05
              ITB-NO,             -- YES --                                 X    BSC06
              RECSEP-1E,          -- TWO HEX DIGITS --                      X    BSC07
              ASCII-NO,           -- YES --                                 X    BSC08
              EBCDIC-YES,         -- NO --                                  X    BSC09
              XPRNCY-NO,          -- YES --                                 X    BSC10
              RESPOL-NO,          -- YES --                                 X    BSC11
              AUTORS-NO,          -- YES --                                 X    BSC12
              INTPOL-NO,          -- YES --                                 X    BSC13
              DA-NO,              -- YES --                                 X    BSC14
              PORT-NO             -- YES --                                      BSC15
        $EBSD TYPE-               -- SEE SYSTEM REFERENCE MANUAL --              BSD00
        $EGEN DSUNIT-$$,          -- R1 / F1 --                             X ++GEN00
              CCUNIT-   ,         -- R1 / F1 / R2 / F2 --                   X   GEN01
              WKUNIT-   ,         -- UNIT / 'UNIT,UNIT,UNIT' --             X   GEN02
              WKPACK-       ,     -- PACK / 'PACK,PACK,PACK' --             X   GEN03
              DIUNIT-##,          -- R1 / F1 / R2 / F2 --                   X ++GEN04
              MINRES-NO,          -- YES --                                 X   GEN05
              DPPACK-       ,     -- PACK NAME --                           X   GEN06
              DPUNIT-             -- UNIT --                                     GEN07
*/*     -- REPLACE THIS CARD WITH /* IN COLUMNS 1-2 --                              00299
*                                                                                  00300
***  PRINT RESULTS OF CCP GENERATION PASS 1                                        00301
*                                                                                  00302
***  IF NO ERRORS, PUNCH INPUT TO CCP GENERATION PASS 2                            00303
*                                                                                  00304
// LOAD  $CC1PP,##                                                                +00305
// FILE NAME-$SOURCE,RETAIN-S,UNIT-##,PACK-PID001,SHARE-NO                        +00306
// RUN                                                                              00307
```

---

[1] For Program Number 5704-SC2 only.

[2] $ECSC and $ECSD are available to give BSCC support (BSCA lines 3 and 4) for Program Number 5704-SC2 only.

Step 3 (User, Card-Oriented):

The user modifies the sample deck to reflect the requirements of his system. Specifically the user can modify:

- The *unit* parameter on the LOAD $MPXDV, unit card to indicate the unit on which the CCP distribution pack resides.
- The *unit* and *pack* parameters on the // FILE of the macro processor OCL statements to indicate the unit and pack on which the work file $SOURCE should be allocated during step 4.
- The CCP $E generation control statements and their operands to indicate the requirements of the system.
- The */* card following the $E control statements. This card is replaced with a /* card.
- The *unit* parameter on the // LOAD $CC1PP, unit card to indicate the unit containing the CCP distribution pack ($CC1PP is the CCP supplied generation utility).
- The *unit* and *pack* parameters on the // FILE statement generation utility OCL statements to indicate the unit and pack where the work file $SOURCE is to be allocated during step 4. The $SOURCE work file will be scratched during *step 6* and the space will be available for other work files.

The user should modify the punched control statements, without inserting any additional cards in the deck, except where he requires more than one:

- $EPLG statement—additional $EPLG statements, with both the LANG and PPUNIT parameters present, are required if more than one programming language is supported.
- $EMLD statement—additional $EMLD statements, with both the TYPE and XMCODE parameters present, are required if more than one type of MLTA terminal is supported.
- $EBSD statement—additional $EBSD statements, with the TYPE parameter present, are required if more than one type of BSC terminal is supported.

Note: If any errors are detected in the user's specifications, the utility program $CC1PP prints them, but does not permit the user to proceed to the next step until those errors are corrected, and the macro processor step repeated. The modified deck, with the initial instruction cards removed, must be placed in the system input device for the next generation step.

Step 4 (System, Card-Oriented):

The system calls the macro processor to analyze and expand the CCP generation $E control statements.

The process creates records in the file $SOURCE. If any errors are detected in the user's specifications, only diagnostic messages from the CCP generation are written to the file. If there are no specification errors, the CCP generation writes to $SOURCE the records necessary to create the specified version of the CCP.

Step 5 (System, Card-Oriented):

The program $CC1PP reads the work file, $SOURCE, and prints what was generated to that file. If there are specification errors, only the user's original statements and the error diagnostic messages are printed. In order to proceed further, the user must correct those errors and perform the macro processor step again.

If there are no errors, the user's original statements are printed. Then the records, which are input for step 6 to create the user's version of the CCP, are printed and punched.

Step 6 (System, Card-Oriented):

The user places the punched output from the previous step, without modification, into the system input device, and begins the major step of the CCP generation. During this step the user's CCP is created. The sequence for this step is:

1. Creating a load module that contains the initial contents of $CCPFILE and the instruction code to initialize that file ($CC1BF).

2. Source generating of $CC4#1, $CC4#2, $CC4VT, and link-editing of the two modules ($CC4#1 and $CC4#2) that make up the resident control program for CCP operation.

3. Link-editing BSC and/or MLTA trace routines.

4. Copying supporting load modules for the operational stage, including startup and shutdown, and for the assignment stage.

5. Creating an initialized module, $CC4Z9, if SECURE-USER operand was specified in the $ESEC generation control statement, later to contain the user's security information.

6. Initializing $CCPFILE, later to be filled with user specifications by an assignment run.

7. Copying the subroutines and macros used in the compilation and link-editing of application programs to run under the CCP.

8. Punching the following cards:

   - OCL and sample generation control statements for an assignment build run, necessary to execute the installation verification program.
   - OCL statements for installation verification program.
   - CGEND.

9. Initializing $CCPDUMP file to be used for trace and storage dumps for 5704-SC2 only (optional).

Note: For Program Number 5704-SC2 only, step 6 follows step 8.

The following is an example of the sample assignment input:

```
$CGSET

*******SAMPLE ASSIGNMENT AND SAMPLE START-UP DECK********************* 00010000
*                                                                       00020000
*********SAMPLE ASSIGNMENT SET ************************************** 00030000
*                                                                       00040000
*****FILL IN UNIT                                                       00050000
*                                                                       00060000
// LOAD $CCPAS,                                                         00070000
*                                                                       00080000
*****FILL IN PACK AND UNIT                                              00090000
*                                                                       00100000
// FILE NAME-$CCPFILE,RETAIN-P,UNIT-  ,PACK-                            00110000
*                                                                       00120000
*****FILL IN PACK AND UNIT                                              00130000
*                                                                       00140000
// FILE NAME-$CCPWORK,RETAIN-S,TRACKS-3,UNIT-  ,PACK-                   00150000
// RUN                                                                  00160000
*                                                                       00170000
*********THE FOLLOWING STATEMENTS CAN BE MODIFIED FOR YOUR              00180000
*********CONFIGURATION BUT SOME MUST BE KEPT TO RUN CCPIVP.  SEE THE    00190000
*********COMMENTS IN THIS DECK.                                         00200000
*                                                                       00210000
// SET ID-a,ACTION-CREATE,DFLTEXEC-YES,ANYSPECS-NO                      00220000
// SYSTEM MINUPA-22K,MINTPBUF-2840,                                     00230000
//       PASSWORD-FECD,                                                 00240000
//       COMMANDL-50,DFFPACK-PROGRAM,PGMREQL-15                         00250000
*                                                                       00260000
// TERMATTR ATTRID-1,TRANSLAT-NO,BLKL-512,DATAFORM-MESSAGE,             00270000
//       VERIFYID-NO,DFF3270-YES                                        00280000
*                                                                       00290000
*********THIS STMNT TYPE REQD FOR CCPIVP(OR MLTALINE STMNT)             00300000
*                                                                       00310000
// BSCALINE TYPE-CS,LINENUM-1,POLLLIST-'00,01,10,11'                    00320000
// BSCATERM TERMID-00,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,    00330000
//       ADDRCHAR-*60604040*,POLLCHAR-*40404040*                        00340000
// BSCATERM TERMID-01,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,    00350000
//       ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*                        00360000
// BSCATERM TERMID-10,TYPE-3277M2,ATTRID-1,COMMAND-NO,                  00370000
//       ADDRCHAR-*61614040*,POLLCHAR-*C1C14040*                        00380000
// BSCATERM TERMID-11,TYPE-3277M2,ATTRID-1,COMMAND-NO,                  00390000
//       ADDRCHAR-*6161C1C1*,POLLCHAR-*C1C1C1C1*                        00400000
*                                                                       00410000
// TERMNAME NAME-CU0DV0,TERMID-00                                       00420000
// TERMNAME NAME-CU0DV1,TERMID-01                                       00430000
// TERMNAME NAME-CU1DV0,TERMID-10                                       00440000
// TERMNAME NAME-CU1DV1,TERMID-11                                       00450000
*                                                                       00460000
*********THIS STMNT TYPE REQD FOR CCPIVP                                00470000
*                                                                       00480000
// DISKFILE NAME-CGIVFIL1,ORG-C,RECL-16                                 00490000
*                                                                       00500000
*********THIS STMNT TYPE REQD FOR CCPIVP                                00510000
*                                                                       00520000
// DISKFILE NAME-CGIVFIL2,ORG-C,RECL-16                                 00530000
*                                                                       00540000
*********NOTE THAT ONE DISKFILE STATEMENT -CGIVFILE- WOULD BE NEEDED    00550000
*********IF SYMBOLIC FILES ARE NOT BEING USED.                          00560000
*                                                                       00570000
* THE FOLLOWING TWO DISK FILE STATEMENTS ARE FOR EXAMPLE ONLY AND       00580000
```

```
*********NOTE THAT ONE DISKFILE STATEMENT -CGIVFILE- WOULD BE NEEDED      00550000
*********IF SYMBOLIC FILES ARE NOT BEING USED.                            00560000
*                                                                         00570000
* THE FOLLOWING TWO DISK FILE STATEMENTS ARE FOR EXAMPLE ONLY AND         00580000
* SHOULD BE REMOVED PRIOR TO RUNNING THIS ASSIGNMENT SET.                 00590000
*                                                                         00600000
// DISKFILE NAME=DUMMY1,ORG=D,RECL=256                                    00610000
// DISKFILE NAME=DUMMY2,ORG=I,RECL=64,KEYL=8,KEYPOS=1,MSTRINDX=YES [1]    00620000
*                                                                         00630000
*********THIS STMNT TYPE REQD FOR CCPIVP IF SYMBOLIC FILES ARE USED.      00640000
*                                                                         00650000
// SYMFILE NAME=CGIVFILE,DISKFILE='CGIVFIL1,CGIVFIL2'                     00660000
*                                                                         00670000
*************THIS STMNT NECESSARY FOR CCPIVP,PACK AND PRINTER VALUES     00680000
**************CAN BE CHANGED FOR YOUR CONFIG.                           00690000
*                                                                         00700000
// PROGRAM NAME=CCPIVP,PGMDATA=YES,                                       00710000
//       FILES='CGIVFILE/CO/NOSHR',                                       00720000
//       PACK=PROGRAM,PRINTER=NO                                          00730000
*                                                                         00740000
*********NOTE THAT CCPIVP MUST BE ON CORRECT PACK AT STARTUP OF CCP.      00750000
*                                                                         00760000
* THE FOLLOWING TWO PROGRAM STATEMENTS ARE FOR EXAMPLE ONLY AND           00770000
* SHOULD BE REMOVED PRIOR TO RUNNING THIS ASSIGNMENT SET.                 00780000
*                                                                         00790000
// PROGRAM NAME=DUMMY1,MRTMAX=2,PGMDATA=YES,                              00800000
// FILES='DUMMY1/DU/SHR,DUMMY2/IRUA/SHR',PACK=SYSTEM,DFEMTERM=4,          00810000
// DFENDF=2,DFFSFDT=1006                                                  00820000
*                                                                         00830000
// PROGRAM NAME=DUMMY2,MRTMAX=2,PGMDATA=YES,                              00840000
// FILES='DUMMY1/DU/SHR,DUMMY2/IRA/SHR',PACK=SYSTEM,DFEMTERM=2,           00850000
// DFENDF=1,DFFSFDT=396                                                   00860000
*                                                                         00870000
*                                                                         00880000
*/*  REPLACE WITH /*                                                    00890000
*                                                                         00900000
*********END OF SAMPLE ASSIGNMENT DECK*********************************  00910000
*                                                                         00920000
*********SAMPLE START-UP OCL FOR CCPIVP*********************************  00930000
*                                                                         00940000
*    FILL IN UNIT                                                         00950000
*                                                                         00960000
// LOAD $CCP,                                                             00970000
*                                                                         00980000
***FOLLOWING TWO // FILE STATEMENTS CORRESPOND TO SAMPLE ASSIGNMENT      00990000
***DECK SYMBOLIC FILES.                                                  01000000
*                                                                         01010000
// FILE NAME=CGIVFIL1,RETAIN=T,TRACKS=1,UNIT=   ,PACK=        ***ANY 5444 01020000
// FILE NAME=CGIVFIL2,RETAIN=T,TRACKS=1,UNIT=   ,PACK=        ***ANY 5444 01030000
*                                                                         01040000
***IF SYMBOLIC FILES ARE NOT USED REPLACE THE PRECEEDING TWO STATEMENTS   01050000
***SINGLE // FILE STATEMENT*************************************          01060000
*                                                                         01070000
// FILE NAME=CGIVFILE,RETAIN=T,TRACKS=1,UNIT=   ,PACK=        ***ANY 5444 01080000
*                                                                         01090000
// RUN                                                                    01100000
*********END OF SAMPLE START-UP OCL*** ************************************ 01110000
```

[1] Not applicable for Program Number 5704-SC2.

To verify that an operational CCP system was generated, the user should:

1. Modify the parameters necessary in the OCL statements and sample assignment control statements.

2. Perform an assignment run that specifies the necessary environmental information to execute the installation verification program (optional).

3. Start the CCP and execute the installation verification program.

Step 1 (User, Cardless-Oriented):

The user enters the following statements from the system input device:

```
/&
// LOAD $MAINT,dsunit
// RUN
// COPY FROM-diunit,TO-PRINT,LIBRARY-S,
   NAME-$CG1.ALL
// COPY FROM-diunit,TO-PRINT,LIBRARY-P,
   NAME-$CG1.ALL
// END
```

where dsunit in the // LOAD statement is the unit on which DSM resides. The diunit in the // COPY statement is the unit on which the CCP distribution pack resides.

Step 2 (System, Cardless-Oriented):

The system retrieves the library members named $CG1.ALL from the source and procedure library on the CCP distribution pack. The system then prints these members. There are three sections to the information printed:

- CCP cardless generation instructions to modify the supplied procedures to reflect your system requirements (member name $CG1GI).

- Sample $E macro statements entered directly from the source library to generate the CCP system (member name $CG1GM) or via the system input device.

- Procedures that perform the CCP generation (member names $CG1G1, $CG1G3, $CG1G4, and $CG1G5).

An example of the printed source and procedure members is:

```
$CG1GI
*************************************************************************** 00000
*              CARDLESS CCP GENERATION INSTRUCTIONS                       * 00010
*************************************************************************** 00020
*                                                                         * 00030
* THE CARDLESS CCP GENERATION PROCEDURE MAKES THE FOLLOWING ASSUMPTIONS:  * 00040
*                                                                         * 00050
*  . THAT DSM RESIDES ON F1                                               * 00060
*                                                                         * 00070
*  . THAT CCP DISTRIBUTION PACK IS ON R1                                  * 00080
*                                                                         * 00090
*  . THAT THE LAST 20 TRACKS ON THE CCP DISTRIBUTION PACK ON R1 ARE       * 00100
*    FREE FOR THE $SOURCE FILE USED BY THE MACRO PROCESSOR                * 00110
*                                                                         * 00120
*  . THAT THE $E MACRO STATEMENTS WILL BE READ FROM THE SOURCE            * 00130
*    LIBRARY MEMBER, $CG1GM, ON THE CCP DISTRIBUTION PACK                 * 00140
*                                                                         * 00150
* IF ANY OF THE ABOVE ASSUMPTIONS ARE NOT TRUE, USE THE MODIFY FUNCTION   * 00160
* OF $MAINT TO CHANGE THE SUPPLIED GENERATION PROCEDURES TO REFLECT THE   * 00170
* REQUIREMENTS OF YOUR SYSTEM.                                            * 00180
*                                                                         * 00190
* SUPPLIED GENERATION PROCEDURES:                                         * 00200
*                                                                         * 00210
*   $CG1G1    *** CARDLESS GENERATION CONTROL                             * 00220
*   $CG1G3    *** PROCESS USER SPECIFICATIONS                             * 00240
*   $CG1G4    *** PRINT RESULTS OF GENERATION SPECIFICATIONS              * 00250
*   $CG1G5    *** CREATE SOURCE AND PROCEDURE MEMBERS                     * 00260
*                                                                         * 00270
* YOU MAY MAKE FOUR KINDS OF MODIFICATIONS TO THE SUPPLIED PROCEDURES:    * 00280
*                                                                         * 00290
*  1. IN THOSE STATEMENTS MARKED ++ IN POSITION 74-75 YOU MAY CHANGE      * 00300
*     THE DSM PACK LOCATION FROM F1 TO R1.                                * 00310
*                                                                         * 00320
*  2. IN THOSE STATEMENTS MARKED ## IN POSITION 72-73 YOU MAY CHANGE      * 00330
*     THE CCP DISTRIBUTION PACK LOCATION FROM R1 TO F1, R2 OR F2.         * 00340
*                                                                         * 00350
*  3. IN THOSE STATEMENTS MARKED $$ IN POSITION 74-75 YOU MAY CHANGE      * 00360
*     THE $SOURCE FILE LOCATION FROM R1 TO F1, R2 OR F2.                  * 00370
*                                                                         * 00380
*  4. THE '// COMPILE' STATEMENT MAY BE REMOVED FROM PROCEDURE $CG1G3     * 00390
*     TO SPECIFY THAT THE MACRO PROCESSOR WILL READ THE $E MACRO          * 00400
*     STATEMENTS FROM THE SYSTEM INPUT DEVICE.                            * 00410
*     -- STATEMENT 03070 --                                               * 00420
*                                                                         * 00430
*  NOTE -- IF A SOURCE MEMBER IS USED IT MUST BE PREPARED PRIOR TO        * 00440
*          CALLING THE PROCEDURE $CG1G1.                                  * 00450
*                                                                         * 00460
* WHEN YOU HAVE MADE THESE CHANGES;                                       * 00470
*   CALL PROCEDURE $CG1G1 BY ENTERING THE FOLLOWING OCL FROM THE          * 00480
*   SYSTEM INPUT DEVICE.                                                  * 00490
*                                                                         * 00500
*   // CALL $CG1G1,UNIT  -- CCP DISTRIBUTION PACK --                      * 00510
*   // RUN                                                                * 00520
*                                                                         * 00530
****** END OF INSTRUCTIONS ******************** END OF INSTRUCTIONS ****** 00540
```

```
$EFAC ESCAPE-'//////',    -- 'CCCCCC' / X'XXXXXXXXXXX' --          X00010000
      PGMCNT-YES,         -- NO --                                 X00020000
      FORMAT-YES,         -- NO --                                 X00030000
      PRUF-NO,            -- YES --                                X00040000
      ACCEPT-NO,          -- YES --                                X00050000
      RESOPN-NO,          -- YES --                                X00060000
      BSYPRT-NO,          -- YES --                                X00061000
      MOVTNT-NO,          -- YES --                                X00062000
      MOVDFF-NO,          -- YES --                                X00063000
      RESREQ-NO,          -- YES --                                X00064000
      SIOC-NO,            -- YES --                                X00065000
      OPPRUF-NO,          -- YES --                                X00066000
      CPUMSG-NO,          -- YES --                                X00067000
      LOWCAS-NO,          -- PF1 - PF12 --                         X00068000
      TTASK-NO            -- YES --                                 00069000
$EPLG LANG-RPGII,         -- COBOL / RPGII / FORTRAN / ASSEM --    X00070000
      PPUNIT-R2           -- R1 / F1 / R2 / F2 --                   00080000
$EPLG LANG-COBOL,         -- COBOL / RPGII / FORTRAN / ASSEM --    X00090000
      PPUNIT-R2           -- R1 / F1 / R2 / F2 --                   00100000
$ESEC SECURE-CCP          -- CCP / USER --                          00110000
$EFIL SETS-3,             -- 1 - 25 --                             X00120000
      PROGS-15,           -- 1 - 999 --                            X00130000
      DFILES-5,           -- 1 - 192 --                            X00140000
      TERMS-5,            -- 2 - 254 --                            X00150000
      DUMPS-2,            -- 2 - 99 --                             X00160000
      CORE-96K,           -- 128K/160K/192K/224K/256K/384K/512K    X00170000
      FLPACK-R2R2R2,      -- NAME OF PACK --                       X00180000
      FLUNIT-R2,          -- R1 / F1 / R2 / F2 --                  X00190000
      DPTRAC-6            -- 0 - 1000 --                            00195000
$EBSC BSCA-1,             -- 0 - 2 --                              X00200000
      DA-NO,              -- YES --                                X00210000
      DIAL-NO,            -- YES --                                X00220000
      PP-NO,              -- YES --                                X00230000
      MP-NO,              -- YES --                                X00240000
      CS-YES,             -- NO --                                 X00250000
      GETMSG-YES,         -- NO --                                 X00260000
      ITB-NO,             -- YES --                                X00270000
      RECSEP-1E,          -- TWO HEX DIGITS --                     X00280000
      ASCII-NO,           -- YES --                                X00290000
      EBCDIC-YES,         -- NO --                                 X00300000
      XPRNCY-NO,          -- YES --                                X00310000
      RESPOL-NO,          -- YES --                                X00320000
      INTPOL-YES,         -- NO --                                 X00330000
      AUTORS-NO,          -- YES --                                X00340000
      PORT-NO             -- YES --                                 00345000
$EBSD TYPE-3277M2         -- SEE SYSTEM REFERENCE MANUAL --         00350000
$ECSC BSCC-1,             -- 0 - 2 --                              X00360000
      GETMSG-YES,         -- NO --                                 X00370000
      ITB-NO,             -- YES --                                X00380000
      RECSEP-1E,          -- TWO HEX DIGITS --                     X00390000
      ASCII-NO,           -- YES --                                X00400000
      EBCDIC-YES,         -- NO --                                 X00410000
      XPRNCY-NO,          -- YES --                                X00420000
      INTPOL-YES,         -- NO --                                 X00430000
      PORT-NO,            -- YES --                                X00432000
      PP-NO,              -- YES --                                X00434000
      CS-YES              -- NO --                                  00436000
$ECSD TYPE-3277M2         -- SEE SYSTEM REFERENCE MANUAL --         00440000
$EGEN DSUNIT-F1,          -- R1 / F1 --                            X00450000
      CCUNIT-R2,          -- R1 / F1 / R2 / F2 --                  X00460000
      WKUNIT-'R2,R1,R1',  -- 'UNIT,UNIT,UNIT'                      X00470000
      WKPACK-'R2R2R2,PID001,PID001', -- 'PACK,PACK,PACK' --        X00480000
      DIUNIT-R1,          -- R1 / F1 / R2 / F2 --          CCP-PID X00490000
      MINRES-NO,          -- YES --                                X00500000
      CARD-NO,            -- YES --                                X00510000
      DPPACK-D2D2D2,      -- PACK NAME --                          X00510001
      DPUNIT-D2           -- UNIT --                                00510002
```

[1] Program Number 5704-SC2 only.

```
$CG1G1
// LOG PRINTER
// NOHALT
*
***  CCP GENERATION CARDLESS PROCEDURE  ***
*
// CALL $CG1G3,R1          *** PROCESS USER SPECIFICATIONS ***
// CALL $CG1G4,R1          *** PRINT RESULTS OF SPECIFICATIONS ***
// CALL $CG1G5,R1          *** CREATE SOURCE AND PROCEDURE MEMBERS ***
// CALL $CCPSA,R1          *** COMPLETE THE CCP GENERATION ***
```

```
$CG1G3
*
***  CCP CARDLESS GENERATION PROCEDURE  ***
***  EXPAND USERS CCP SPECIFICATIONS    ***
*
// LOAD  $MPXDV,R1
// FILE  NAME-$SOURCE,RETAIN-T,UNIT-R1,PACK-PID001,TRACKS-20,
//       LOCATION-386
// COMPILE SOURCE-$CG1GM,UNIT-R1
// RUN
*
*
*  NOTE:
*  -----
*
*  $E MACRO STATEMENTS ARE ENTERED VIA THE SOURCE LIBRARY MEMBER $CG1GM
*  ON THE CCP DISTRIBUTION PACK OR FROM THE SYSTEM INPUT DEVICE IF THE
*  '// COMPILE' STATEMENT IS REMOVED FROM THIS PROCEDURE.
```

```
$CG1G4
*
***  CCP CARDLESS GENERATION PROCEDURE  ***
*
***  PRINT RESULTS OF CCP GENERATION SPECIFICATIONS
*
// LOAD  $CC1PP,R1
// FILE NAME-$SOURCE,UNIT-R1,PACK-PID001
// RUN
```

```
$CG1G5
*
***  CCP CARDLESS GENERATION PROCEDURE  ***                                    05010
*                                                                              05020
***  IF NO ERRORS IN GENERATION SPECIFICATIONS, LOAD $MAINT TO CREATE          05030
***  SOURCE AND PROCEDURE MEMBERS THAT WILL COMPLETE THE CCP GENERATION        05040
*                                                                              05050
// LOAD $MAINT,F1                                                              05060
// FILE NAME-$SOURCE,UNIT-R1,PACK-PID001,RETAIN-S                            ++05070
// RUN                                                                      ##$$05080
// COPY FROM-DISK,TO-R1,RECL-96,RETAIN-R,FILE-$SOURCE                          05090
// END                                                                      ##$$05100
                                                                               05110
```

Step 3 (User, Cardless-Oriented):

The user modifies the sample procedures to reflect the requirements of the user's system. This is done by using the MODIFY statement in the utility program $MAINT. (For a description of the MODIFY statement, see the *IBM System/3 Model 15 System Control Programming Reference Manual*, GC21-5077 or *IBM System/3 Model 15 System Control Program Concepts and Reference Manual*, GC21-5162.) Specifically the user can modify:

- The DSM pack location.
- The CCP distribution pack location.
- The unit, pack and location for the $SOURCE file required by the macro processor.
- The source member name on the // COMPILE statement to reflect which source member contains the user's CCP generation specifications.
- The // COMPILE statement can be removed from procedure $CG1G3 to specify that, during step 4, the macro processor reads the user specification from the system input device instead of from a source library member.

Note: If the // COMPILE statement is used in procedure $CG1G3, the source member described must be prepared prior to calling procedure $CG1G1. See step 4 for a description of entering the $E control statements.

Step 4 (User/System, Cardless-Oriented):

The user calls procedure $CG1G1 from the CCP distribution pack. This procedure controls the remaining portion of the CCP generation. It does this by making calls to these procedures: $CG1G3, $CG1G4, $CG1G5, and $CCPSA respectively. Each of these procedures performs a different generation process. Procedure $CG1G3 is described in this step. Procedures $CG1G4 and $CG1G5 are described in step 5; procedure $CCPSA is described in step 6. Procedure $CCPSA is a procedure generated during step 5, based on the user's input, that controls the generation of the user's CCP pack.

Procedure $CG1G3 loads the macro processor and allows the user to enter the $E control statements. There are two ways the user can do this: via the system input device or from a source library member. However, to enter input from the system input device, the user must remove the // COMPILE statement from this procedure. (For a description of the // COMPILE statement, see the *IBM System/3 Model 15 System Control Programming Reference Manual*, GC21-5077 or *IBM System/3 Model 15 System Control Program Concepts and Reference Manual*, GC21-5162.)

The source member, $CG1GM, printed in step 1 can be used as a guide for entering the $E control statements. This module can also be modified for use as input to the macro processor since the // COMPILE statement in procedure $CG1G3 specifies $CG1GM as the input source member.

The user should enter the control statements without inserting any additional statements except where more than one is required:

- $EPLG statement — additional $EPLG statements, with both the LANG and PPUNIT parameters present, are required if more than one programming language is to be supported.
- $EMLD statement — additional $EMLD statements, with both the TYPE and XMCODE parameters present, are required if MLTA terminals are to be supported.
- $EBSD statement — additional $EBSD statements, with the parameter TYPE present, are required if more than one type of BSCA terminal is to be supported.
- $ECSD statement (Program Number 5704-SC2 only) — additional $ECSD statements, with the parameter TYPE present, are required if more than one type of BSCC terminal is to be supported.

Step 5 (System/User, Cardless-Oriented):

Procedure $CG1G4 prints the results of the CCP generation specifications. If there are specification errors, the user's statements and the error diagnostic messages are printed. Before continuing, the user must restart the CCP generation at procedure $CG1G1 and provide error free specifications.

If there are no errors, the user's statements are printed. Then procedure $CG1G5 is called and the records which will be input to step 6 are printed. Procedure $CG1G5 loads $MAINT to create the source and procedure members that will complete the CCP generation. Also printed is the $SOURCE file setup for the $MAINT file-to-library run.

Step 6 (System, Cardless-Oriented):

The system calls the generated procedure $CCPSA from the distribution pack and begins the major step of the CCP generation. During this step the user's CCP is created. The sequence for this step is:

1. Creation of a load module which contains the initial contents of $CCPFILE and the instruction code to initialize that file ($CC1BF).

2. Source generation and link edit of the modules $CC4#1 and $CC4#2, which is the resident control program during CCP operation.

3. Copying of supporting load modules for the operational stage, including startup, shutdown, and the assignment stage.

4. Creation of an initialized module $CC4Z9 if SECURE-USER was specified in the $ESEC statement; this module will contain the user's security information.

5. Initialization of $CCPFILE and $CCPDUMP (5704-SC2 only) later to be filled with user specifications by assignment runs.

6. Copying of the subroutines and macros used in the compilation and link edit of application programs to run under the CCP.

7. Print the sample control statements for an assignment build run necessary to execute the installation verification program. On the following page is an example of the sample assignment input.

   Note: For Program Number 5704-SC2 only, step 5 follows step 7.

```
$CGSST
******   SAMPLE ASSIGNMENT SET CONTROL STATEMENTS     ******                00010000
*                                                                           00020000
******   THE FOLLOWING STATEMENTS CAN BE MODIFIED FOR YOUR                  00030000
******   CONFIGURATION BUT SOME MUST BE KEPT TO RUN CCPIVP.                 00040000
*                                                                           00050000
*                                                                           00060000
// SET ID-a,ACTION-CREATE,DFLTEXEC-YES,ANYSPECS-NO                          00070000
// SYSTEM MINUPA-22K,MINTPBUF-2840,                                         00080000
// PASSWORD-FECD,                                                           00090000
// COMMANDL-50,DFFPACK-PROGRAM,PGMREQL-15                                   00100000
*                                                                           00110000
// TERMATTR ATTRID-1,TRANSLAT-NO,BLKL-512,DATAFORM-MESSAGE,                 00120000
// VERIFYID-NO,DFF3270-YES                                                  00130000
*                                                                           00140000
******   THIS STMNT TYPE REQD FOR CCPIVP(OR MLTALINE STMNT)                00150000
// BSCALINE TYPE-CS,LINENUM-1,POLLLIST-'00,01,10,11'                        00160000
// BSCATERM TERMID-00,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,        00170000
// ADDRCHAR-*60604040*,POLLCHAR-*40404040*                                  00180000
// BSCATERM TERMID-01,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,        00190000
// ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*                                  00200000
// BSCATERM TERMID-10,TYPE-3277M2,ATTRID-1,COMMAND-NO,                      00210000
// ADDRCHAR-*61614040*,POLLCHAR-*C1C14040*                                  00220000
// BSCATERM TERMID-11,TYPE-3277M2,ATTRID-1,COMMAND-NO,                      00230000
// ADDRCHAR-*6161C1C1*,POLLCHAR-*C1C1C1C1*                                  00240000
*                                                                           00250000
// TERMNAME NAME-CU0DV0,TERMID-00                                           00260000
// TERMNAME NAME-CU0DV1,TERMID-01                                           00270000
// TERMNAME NAME-CU1DV0,TERMID-10                                           00280000
// TERMNAME NAME-CU1DV1,TERMID-11                                           00290000
*                                                                           00300000
******   THIS STMNT TYPE REQD FOR CCPIVP                                   00310000
// DISKFILE NAME-CGIVFILE,ORG-C,RECL-16                                     00320000
*                                                                           00330000
******   THIS STMNT NECESSARY FOR CCPIVP                                   00340000
// PROGRAM NAME-CCPIVP,PGMDATA-YES,PRINTER-SHR,                             00350000
// FILES-'CGIVFILE/CO/NOSHR',                                               00360000
// PACK-PROGRAM                                                             00370000
```

To Verify Generation (User, Cardless-Oriented)

To verify that an operational CCP system has been generated, the user should:

1. Modify the sample assignment set contained in the source library member $CGSST on the CCP production pack. To do this, use the MODIFY statement in the utility program $MAINT. (For information on the MODIFY statement, see the appropriate SCP reference manual listed under ***Related Publications*** in the Preface.)

2. Perform an assignment run which specifies the necessary environmental information to execute the installation verification program.

3. Start the CCP and execute the installation verification program.

GENERATION CONTROL STATEMENTS

Each CCP generation requires a set of generation control statements. If the required statements are not specified, no generation takes place. Some generation statements are always required for CCP support, and others are required only if the user desires a certain option.

Those control statements always required are:

$EFAC	CCP facilities
$EPLG	Programming languages
$EFIL	$CCPFILE allocation
$EGEN	CCP generation stream

The optional control statements are:

$ESEC	Terminal sign-on security	
$EMLA	MLTA support	
$EMLD	MLTA devices	
$EBSC	BSC support	
$EBSD	BSC devices	
$ECSC	BSCC support	5704-SC2 only
$ECSD	BSCC support	

Optional statements must be present to include support for MLTA, BSCA, or BSCC; that is, the $EMLA statement and at least one $EMLD statement must be present for MLTA support, the $EBSC statement and at least one $EBSD statement must be present for BSCA support, and the $ECSC statement and at least one $ECSD statement must be present for BSCC support (5704-SC2 only).

All generation control statements (if present) must be in the following order:

$EFAC
$EPLG
$ESEC
$EFIL
$EMLA[1]
$EMLD[1]
$EBSC
$EBSD
$ECSC } 5704-SC2 only
$ECSD }
$EGEN

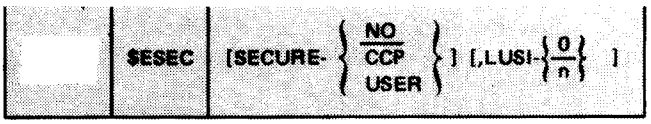
[1] MLTA and BSCC are mutually exclusive.

Writing Generation Control Statements

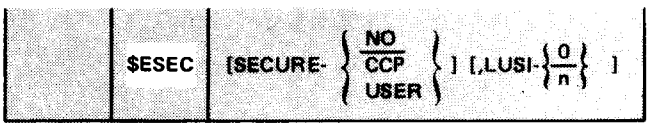
Starting in Column

1	8	14	72
Name	Operation	Operands	Continuation
Symbol or blank	Statement name	No operands or one or more separated by commas	Any nonblank character if continuation is being used

The name field can contain any valid assembly language symbolic name beginning in column 1. The name is assigned to the first byte of generated code. *Because the name is optional, it is not shown.*

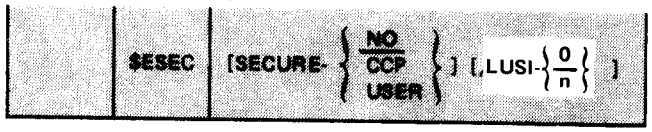


The desired mnemonic operation code (control statement name) must appear as specified in the control statement description. The operation code must start in column 8.

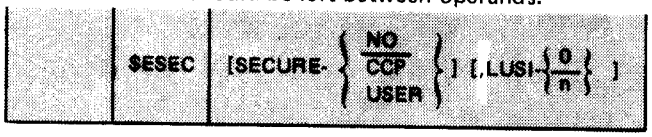


The operands specify available services and options. The operands must start in column 14 and are written as follows:

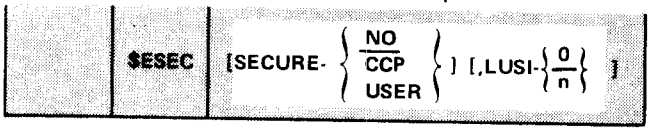
1. Each operand consists of a keyword followed by a hyphen and a parameter.



2. No blanks should be left between operands.

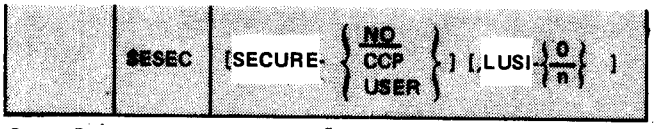


3. Commas *precede* all but the first operand.

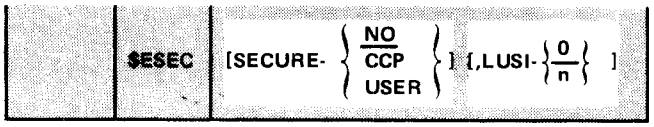


4. The parameter part of the operand must immediately follow the dash.

5. The keyword part of each operand must correspond to one of the keywords in the control statement description.



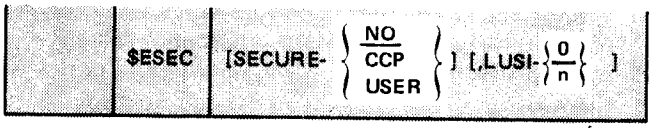
6. Some operands are not required. These optional operands are indicated by enclosing the operand within brackets [KEYWORD-parameter]. The operand enclosed in the brackets can be coded if the associated option is desired. The bracket symbols [] are used to help define the control statements. These symbols are not coded; they are only used to indicate how a control statement can be written.



7. An option list for a keyword parameter is specified as follows:

$$\left\{\begin{matrix}\text{NO}\\\text{CCP}\\\text{USER}\end{matrix}\right\}$$

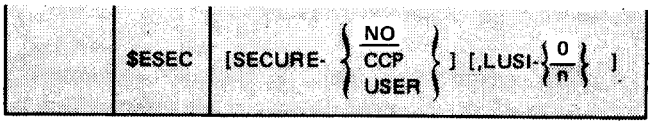
Braces { } indicate that a choice must be made. One of the parameters from the vertical stack within braces must be coded, depending on which of the associated services is desired. The symbols { } are used to help define the control statements. These symbols are not coded; they are only used to indicate how a control statement can be written.



8. The operands can be written in any order. If a keyword is not specified, the default value is used. A default value is selected for optionai keywords that are omitted. The default value is indicated in the macro instruction description by a line under the default option. For example,

$$\text{KEY-}\left\{\begin{matrix}\underline{\text{A}}\\\text{B}\\\text{C}\end{matrix}\right\}$$

indicates that option A is the default value.



9. No operands can be specified beyond column 71. If continuation is required, column 72 must contain a nonblank character and the last operand must be followed by a comma. An operand cannot be divided and continued on the next line. The operands of the continued field must begin in column 14. For an example of continuation coding, see Figure 4.

10. Comments must be separated from the operand or comma by at least one blank space. Comments cannot be inserted between operands on a one-line control statement. Figure 5 shows examples of comments used with control statements. On the assembly listing, all comments on the generated code are justified by the macro processor to begin in column 40. Any comments too long to be contained in columns 40 through 71 are truncated from the right.

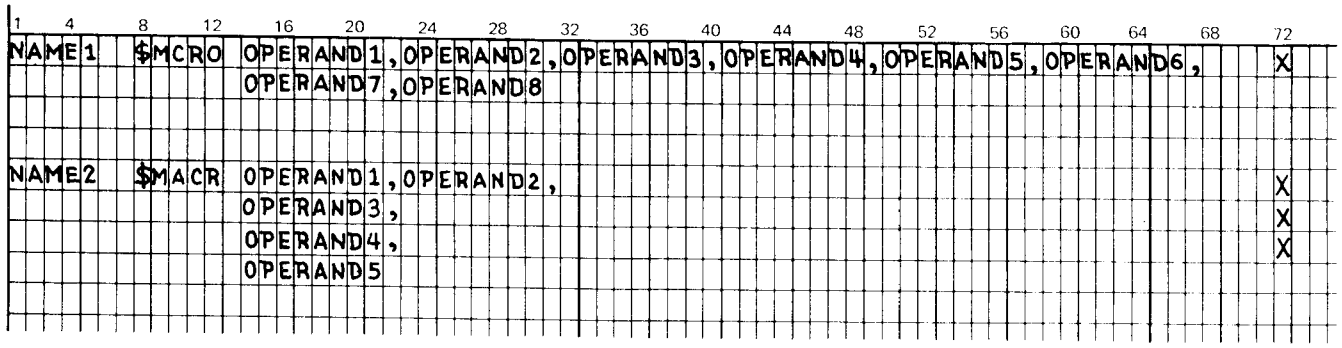


Figure 4. Continuation Coding Examples

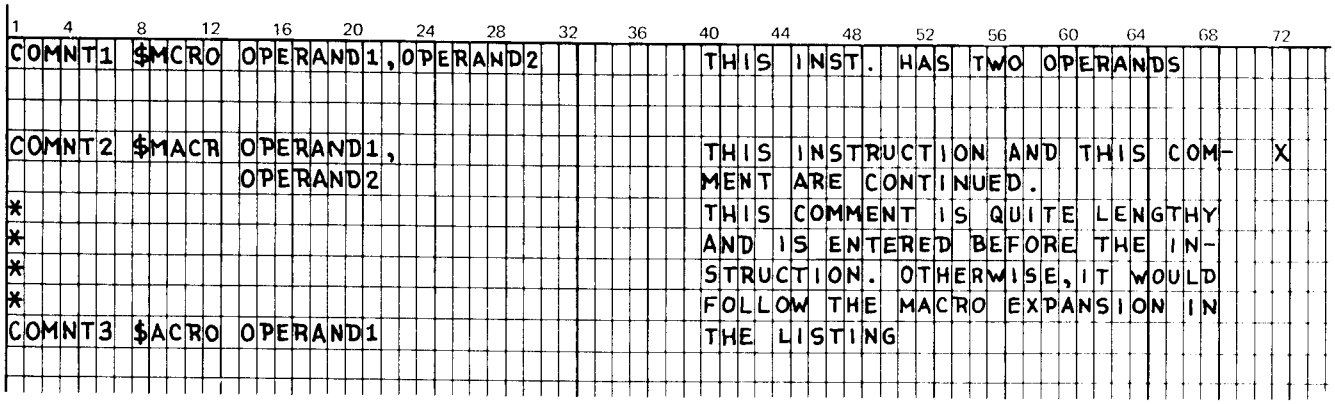


Figure 5. Comments on Macro Instructions

Most operands have default values. The user can have a control statement present and have all operands absent if he accepts the default values.

In all cases where YES is appropriate, the single letter Y can be used. In all cases where NO is appropriate, the single letter N can be used.

In the following generation control statement descriptions the term *object system* refers to the system that executes the CCP after generation.

What You Must Know

- _____ Whether the data mode escape facility is to be generated, and if so the six user-specified data mode escape characters.
- _____ Will a program-request count be kept?
- _____ Is display format facility (DFF) supported?
- _____ Is program request under format (PRUF) supported?
- _____ Is core resident accept input supported?
- _____ Is resident OPEN/CLOSE supported?
- _____ Is 328X printer busy support to be included?
- _____ Is DFF to be moved out to support more terminals?
- _____ Are the Terminal Name Tables (TNT) to be moved out to support additional terminals?
- _____ Is resident program request supported?
- _____ Is the serial I/O channel RPQ to be supported?
- _____ Is the Task-to-Task facility to be supported?

The $EFAC generation control statement indicates options that determine the CCP facilities to be included during generation.

This statement is always required and must be the first of the generation control statements. Only one $EFAC statement can be entered. All operands of this statement are optional.

```
$EFAC
      [,ESCAPE-  { NO          }          ]
                 { 'cccccc'    }
                 { X'xxxxxxxxxxxx' }
      [,PGMCNT-  { YES }       ]
                 { NO  }
      [,FORMAT-  { YES }       ]
                 { NO  }
      [,PRUF-    { YES }       ]
                 { NO  }
      [,OPPRUF-  { YES }       ] (5704-SC2 only)
                 { NO  }
      [,CPUMSG-  { NO  }       ] (5704-SC2 only)
                 { YES }
      [,ACCEPT-  { YES }       ]
                 { NO  }
      [,RESOPN-  { YES }       ] (5704-SC2 only)
                 { NO  }
      [,BSYPRT-  { YES }       ]
                 { NO  }
      [,MOVDFF-  { YES }       ] (5704-SC2 only)
                 { NO  }
      [MOVTNT-   { YES }       ] (5704-SC2 only)
                 { NO  }
      [,RESREQ-  { YES }       ] (5704--SC2 only)
                 { NO  }
      [,SIOC-    { YES }       ] (5704-SC2 with
                 { NO  }         5799-WNK)
      [,LOWCAS-  { NO   }      ] (5704-SC2 only)
                 { PF1  }
                 { PF2  }
                 {  :   }
                 { PF12 }
      [,TTASK-   { YES }       ] (5704-SC2 only)
                 { NO  }
```

ESCAPE- $\left\{\begin{matrix}\underline{NO} \\ 'cccccc' \\ X'xxxxxxxxxxxx'\end{matrix}\right\}$

The ESCAPE operand specifies whether data mode escape is supported and if so, the six data mode escape characters to be used. The data mode escape function is used by the terminal operator to instruct the CCP to accept the next input as a command to the CCP and not as data for the program with which the terminal had been communicating. This is the only way a terminal can interrupt the normal course of a program it requested. The CCP checks all data input from a requesting terminal for this string of characters as the first six bytes.

Be sure this string is not a sequence of bytes that can inadvertently be entered as data. It is suggested that the string be made up of six special characters such as slash (/).

EBCDIC characters or hex digits can be used for this value. If EBCDIC characters are used, this value must be made up of exactly six characters. It is suggested the value be enclosed in apostrophes as shown, because certain characters chosen for this string might be considered operand delimiters by the macro processor.

If the apostrophe (') is to be one of the characters of the string, then each such apostrophe must be coded as four successive apostrophes, and the parameter must be bounded by apostrophes.

If the hex form is used, the parameter must be coded as:

- The letter X
- A single apostrophe (')
- Exactly 12 hex digits
- A single apostrophe (')

The default value is NO, indicating no terminal can interrupt a program and communicate directly with the CCP. For a minimum system this value is NO.

PGMCNT- $\left\{\begin{matrix}YES \\ \underline{NO}\end{matrix}\right\}$

The PGMCNT operand specifies whether a count should be kept of the number of times a user program was requested. Not all requests for a user program are counted. The exceptions include requests for an already active MRT and certain rejected requests. The count is a request count, not an execution count. YES indicates these counts are to be accumulated during the execution of the CCP, and are to be added to previous counts in $CCPFILE during the CCP shutdown. The default is NO, indicating program-request counts are not required.

These counts can be used to design or redesign user programs to take advantage of certain CCP features (such as reusability or multiple requesting terminals) affecting program request response time.

For Program Number 5704-SC2, if the active assignment set has been changed since startup, the counts will not be updated.

For a minimum system the value is NO.

FORMAT- $\left\{\begin{matrix}YES \\ \underline{NO}\end{matrix}\right\}$

The FORMAT operand specifies whether the display format facility (DFF) is to be included in the CCP. A value of YES includes DFF support in the CCP. The default value is NO indicating a CCP system without DFF support is to be generated.

PRUF- $\left\{\begin{matrix}YES \\ \underline{NO}\end{matrix}\right\}$

The PRUF operand specifies whether program request under format (PRUF) is supported. YES must be specified if using the PRUFLNG parameter in the PROGRAM statement at assignment time. The default is NO. For more information on PRUF, see the *IBM System/3 Communications Control Program Programmer's Reference Manual,* GC21.7579.

OPPRUF- {YES / NO} (5704-SC2 only)

The OPPRUF operand specifies whether PRUF is to optimize the TP buffer for PRUF mode terminals. If YES is specified, CCP will invite command mode PRUF terminals only for the data expected from that terminal. (DFF calculates the amount of data that is to be received from a terminal.) If NO is specified, the command mode PRUF terminals will always be invited for the maximum PRUFLNG specified in the assignment set. This function may be needed if CCP is communicating with a System/7 or a Series/1; for example, that is supporting lines in addition to CCP. The default is NO.

CPUMSG- {NO / YES} (5704-SC2 only)

The CPUMSG operand specifies whether information messages (S type) are to be sent to the CPUs. YES indicates that messages are to be sent. YES should be specified if the CPU is a System/34. The default is NO.

ACCEPT- {YES / NO}

The ACCEPT operand specifies whether the resident code will handle the user input data or whether a transient will have to be called. YES indicates the resident code will handle the input; NO indicates a transient will have to be called. The default is NO.

RESOPN- {YES / NO} (5704-SC2 only)

The RESOPN operand specifies whether or not the resident routine is to be incorporated to handle the pseudo open/close/allocate functions of CCP. YES indicates that the resident routine handles the processing and occupies 10K bytes that would otherwise be allocated to the user program area. NO indicates that transients are invoked for this function. The default is NO. If RESOPN is yes for generation, an external pointer list (EPL) is built during start-up time.

BSYPRT- {YES / NO}

The BSYPRT operand specifies whether the code to handle a 328X printer busy condition is to be included in the CCP. YES indicates that the support is to be included. YES allows CCP to detect a printer busy condition, retain control until the printer has completed the operation, and then return control to the application program. The default is NO.

MOVDFF- {YES / NO} (5704-SC2 only)

The MOVDFF operand specifies whether the resident routine that supports DFF is to be moved from $CCP#2 to a 4K-byte area that would otherwise be allocated to the user program area. The purpose of this move is to enable the system to support additional terminals. YES indicates that DFF is moved out. The default is NO. If MOVDFF is YES, DFF must also be YES.

MOVTNT- {YES / NO} (5704-SC2 only)

The MOVTNT operand specifies whether the Terminal Name Tables (TNT) within resident CCP are to be moved to a 2K or 4K-byte area that would otherwise be allocated to the user program area. The purpose of this move is to enable the system to support additional terminals. YES indicates that the TNTs are to be moved out. The default is NO.

RESREQ- {YES / NO}

The RESREQ operand specifies whether the resident routine is to be incorporated to handle the program request functions of CCP. YES indicates that the resident routine handles the processing and occupies 4K-byte area that would otherwise be allocated to the user program area. NO indicates that transients are invoked for this function. The default is NO (5704-SC2 only).

SIOC- {YES / NO} (5704-SC2 with 5799-WNK)

The SIOC operand specifies whether support for channel connected systems is to be included in this generation. The default is NO. (See *IBM System/3 Model 15D Channel Connected Systems Program Reference and Logic Manual*, GC21-5199.)

LOWCAS- {NO / PF1 / PF2 / ⋮ / PF12} (5704-SC2 only)

The LOWCAS operand specifies the PF key to be used to indicate lower case input from a terminal for program request data only. NO indicates that all input will be upper case. The default is NO.

TTASK- { YES / NO } (5704-SC2 only)

The TTASK operand specifies whether support for the task-to-task communication function is to be included in this generation. The default is NO. (See *IBM System/3 Communications Control Program Programmer's Reference Manual,* GC21-7579.)

$EPLG—Programming Languages

What You Must Know

_____ Programming languages to be supported by the CCP.

_____ Disk unit on which the pack used to prepare programs written in that language is mounted during generation.

Each $EPLG generation control statement indicates a programming language for program preparation under CCP. The following rules apply to this statement:

- At least one $EPLG statement is required.
- One $EPLG statement is required for each programming language supported by the CCP. Only one $EPLG statement can be entered for any one language.
- The first $EPLG statement must immediately follow a $EFAC statement. Multiple $EPLG statements can come in any order after the first.
- All operands on this statement are required.

$EPLG	LANG- { COBOL / FORTRAN / ASSEM / RPG II }	,PPUNIT- { R1 / F1 / R2 / F2 }	

LANG- { COBOL / FORTRAN / ASSEM / RPGII }

The LANG operand specifies a programming language that is to be supported by the CCP. This operand indicates the language that is to be included in the CCP. ASSEM is the abbreviation for Basic Assembler language. At least one of the languages is required. There is no default.

PPUNIT- { R1 / F1 / R2 / F2 }

The PPUNIT operand specifies the disk unit containing a pack used for compiling application programs for the CCP. For example, this is the pack containing the RPG II compiler. Use this operand to specify the disk unit for this pack during generation. There is no default. Any 5444 unit is valid, except the unit on which the distribution pack is mounted.

The assembler user gets the CCP $N macros copied into his source library on this pack. He must designate the PPUNIT as a unit on which the Macro Processor resides. The CCP $N macros must be on the same pack as the macro processor or the DSM pack from which the user will perform initial program load (IPL) when assembling CCP programs.

A // PAUSE card containing comments on directions for proceeding is generated into the input stream for copying and/or renaming data management modules. This allows the user to ensure that the correct pack is mounted on the correct unit. The routines copied to the pack on this unit are:

LANG Parameter	What is Copied
COBOL	Communications service subroutine CCPCIO
FORTRAN	Communications service subroutine CCPFIO
ASSEM	Communications service macros $NCOM, $NPLO, $NOPV, $NRTV, $NPL, and $NCIO
RPG II	Communications service subroutines SUBR90, SUBR91, SUBR92, SUBR93, and SUBR88

$ESEC—Terminal Sign-On Security

What You Must Know

_____ What type of sign-on security used, if any.

_____ Length of the user's security information, if a user written sign-on security routine is used.

The $ESEC generation control statement indicates the type of terminal sign-on security used (if any).

This statement is optional. It is included only if terminal sign-on security is desired. If included, this statement must immediately follow a $EPLG statement. Only one $ESEC statement can be entered. All operands of this statement are optional.

$ESEC	[SECURE- {NO / CCP / USER}] [,LUSI- {0 / n}]

SECURE- {NO / CCP / USER}

The SECURE operand specifies whether terminal sign-on security support is wanted. The default is NO. If the default is used, any command terminal can sign on to the user's system without presenting any validation information.

CCP indicates that the CCP password security routines will be included and used. Each command terminal user must then give the proper one-to-six character password as the operand of his sign-on command before he can address the system to use further facilities. The actual password is specified in an assignment run.

USER indicates that the user wishes to include his own sign-on security routines instead of using those of the CCP. These must follow the rules specified for such routines (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579). In addition, at startup the user's security information must exist in a load module named $CC4Z9.

For a minimum system, the value is NO.

LUSI- {0 / n}

The LUSI operand specifies the length (number of bytes) of the user's security information. It should be omitted unless SECURE-USER is also specified. The default value is 0 and the maximum is 4096. However, the value of this operand must not be 0 if SECURE-USER was specified.

If the user specifies SECURE-USER, at startup this number of bytes is moved from load module $CC4Z9 to the user's security work area reserved in the resident CCP supervisor during the CCP generation.

For a minimum system, the value is 0.

$EFIL—$CCPFILE Allocation

What You Must Know

_____ Anticipated number of assignment sets to be placed into $CCPFILE.

_____ Anticipated number of programs and files in an assignment set.

_____ Anticipated number of terminals in your configuration.

_____ Number of main storage dumps to be retained in $CCPDUMP during error situations.

_____ Main storage size of processing unit.

_____ Disk unit and pack name on which $CCPFILE is to be created.

_____ Beginning track location for $CCPFILE, if you wish to position the file on the pack.

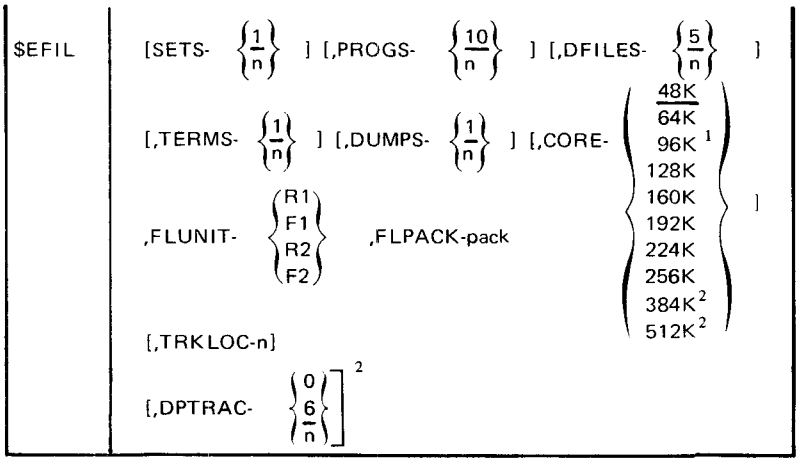
The $EFIL generation control statement provides information that affects the allocated size and location of $CCPFILE.

This statement is always required. Only one $EFIL statement can be entered. This statement must immediately follow the $ESEC statement (if present) or the $EFAC statement (if $ESEC is not present).

All operands, except the FLUNIT, FLPACK, and TRKLOC operands, are used by the CCP generation to estimate the disk space required for $CCPFILE and $CCPDUMP.

You need not be overly concerned about allocating too little space for $CCPFILE or $CCPDUMP. If later you find that the space is too small for your requirements, you can enlarge the space for the file by doing one of the following:

- Use the $COPY program to copy $CCPFILE to a larger space on disk.
- Use the program $CC1BF, which is present on your CCP Production pack, and specify a FILE OCL statement to indicate the number of tracks required. Run $CC1BF; then rerun assignment to reestablish the contents of $CCPFILE.
- Use the program $CC1DP, which is present on your CCP Production pack, and specify a FILE OCL statement to indicate the number of tracks required for the $CCPDUMP file, as well as a // TRACE TRACKS-nn statement to allocate the trace area.
- (5704-SC2 only.) If you anticipate running the assignment build program ($CCPAS) while CCP is in the execution phase, the number of sets anticipated should be increased by one. This will increase the initial disk space allocation of $CCPFILE allowing the CCP startup phase enough space to copy certain tables to the end of the file. These tables are needed by the CCP execution phase. With this arrangement, the rest of $CCPFILE is available for $CCPAS to perform its functions on the file.



[1] Default for Program Number 5704-SC2.
[2] Available as options for Program Number 5704-SC2.

SETS- $\left\{\frac{1}{n}\right\}$

The SETS operand indicates the maximum number of assignment sets that the user anticipates placing into $CCPFILE. Only one assignment set need be defined. If space is available at assignment, it is possible to place more sets into $CCPFILE than the value given here. This value simply serves as a guideline in allocating space for the file.

The default value is 1. For a minimum $CCPFILE the value is 1; the maximum is 25.

PROGS- $\left\{\frac{10}{n}\right\}$

The PROGS operand indicates the number of user programs anticipated for each assignment set in $CCPFILE. The minimum value is 1; the maximum for 5704-SC1 is 255, for 5704-SC2 is 999. The default value is 10.

The value given here does not place any actual restriction on the number of user programs in an assignment set, but simply serves as a guideline to the CCP generation in allocating space for $CCPFILE.

DFILES- $\left\{\frac{5}{n}\right\}$

The DFILES operand indicates the average number of disk files anticipated for each assignment set in $CCPFILE. The minimum is 0. The maximum for Program Number 5704-SC1 is 50; the maximum for Program Number 5704-SC2 is 192. The default value is 5.

The value given here does not place any actual restriction on the number of disk files in an assignment set, but simply serves as a guideline to the CCP generation in allocating space for $CCPFILE.

Another factor to be considered in determining this operand value is the number of // FILE cards that can be entered at the CCP startup (the maximum number of files that can be used in any CCP run). The DSM allowable maximum is 40 for Program Number 5704-SC1, and 192 for Program Number 5704-SC2. If more than the maximum number of files is defined for an assignment set when starting up the CCP, the excess must be suppressed, during startup by the system operator. For a minimum $CCPFILE, a value of 0 can be specified.

TERMS- $\left\{\frac{1}{n}\right\}$

This optional operand indicates the number of terminals anticipated for each assignment set in $CCPFILE. The minimum is 1 (the default value) and the maximum is 254.

The value given here does not place any actual restriction on the number of terminals in an assignment set, but simply serves as a guideline to the CCP generation in allocating space for $CCPFILE.

For a minimum $CCPFILE this value is 1.

DUMPS- $\left\{\frac{1}{n}\right\}$

The DUMPS operand indicates space is to be reserved in $CCPFILE (or $CCPDUMP for 5704-SC2) for this number of dynamic main storage dumps. The default value is 1. The minimum value is 1, and the maximum value is 9. For Program Number 5704-SC2, the maximum value is 99.

When a user program is abnormally terminated by the CCP and space is available in $CCPFILE (or $CCPDUMP for 5704-SC2) for dumps, the supervisor and the program partition in which CCP is running, are written to disk. If space for a complete dump is not available, no dump is made. For a discussion of the dump to disk from forcible termination, see *IBM System/3 Model 15 Communications Control Program System Operator's Guide,* GC21-7619. If the non-CCP program partition size is not equal to zero, more than the specified number of dumps can be taken. If $CCPDD is included in the assignment set (only necessary for 5704-SC2) $CCPDD can be used to dump the dump area while CCP is executing, thereby making room for more main storage dumps.

CORE- { 48K[3] | 64K[3] | 96K[1] | 128K | 160K | 192K | 224K | 256K | 384K[2] | 512K[2] }

The CORE operand indicates processing unit size. The default and minimum values are 48K for Program Number 5704-SC1, and 96K for Program Number 5704-SC2. This operand value indicates the total main storage size, not the size of the anticipated CCP program partition.

$CCPFILE (or $CCPDUMP for 5704-SC2) dynamic main storage dump space allocation is based on this value. At startup, the DSM configured size of the processing unit running the CCP is used to determine how many core dumps can be taken during the run.

FLUNIT- { R1 | F1 | R2 | F2 }

The FLUNIT operand specifies the unit on which, during generation, the unit where $CCPFILE is to be allocated. There is no default parameter.

The value specified here need not be the CCP production pack.

During the second pass of the CCP generation, when the space for $CCPFILE is about to be allocated and its contents initialized, a // PAUSE statement, which is provided in the input, permits the system operator to verify that the correct pack is mounted on this unit and, if not, to mount it.

FLPACK-pack

The FLPACK operand specifies the pack name upon which $CCPFILE is to be allocated. There is no default value.

[1] Default for Program Number 5704-SC2.
[2] Available for Program Number 5704-SC2.
[3] Available for Program Number 5704-SC1 only.

TRKLOC-n

The TRKLOC operand can be used to specify the beginning track location for the allocation of $CCPFILE.

If this operand is omitted, $CCPFILE is allocated in the location determined by the DSM.

DPTRAC-n

This optional operand indicates the number of tracks that are to be allocated for the CCP trace area within the CCP dump file ($CCPDUMP). The default is 0; the maximum is 1000 (5704-SC2 only). The minimum non-zero value that is allowed for trace is 6 tracks.

$EMLA—MLTA Support

What You Must Know

_____ Number of MLTA lines to be supported.

_____ If MLTA is to be supported, whether all programs will use translated (EBCDIC) data, or whether some programs will process data in line transmission code.

The $EMLA generation control statement indicates whether the Multiple Line Terminal Adapter (MLTA) is to be used by the CCP and, if so, the number of MLTA lines.

This statement is optional. It need be included only if the MLTA is to be supported. Either the MLTA statements ($EMLA and $EMLD), the BSCA statements ($EBSC and $EBSD), or the BSCC statements ($ECSC and $ECSD) must be included. Both BSCA and MLTA can be specified. However, BSCC and MLTA are mutually exclusive. Only one $EMLA statement can be entered. If included:

- This statement must immediately follow the $EFIL statement.
- At least one $EMLD statement is required to immediately follow this statement if the value of operand LINES is not 0.

$EMLA	LINES-n[,XLATE- { YES | NO }]

LINES-n

The LINES operand indicates the number of teleprocessing lines on the user's MLTA. The minimum is 0, and the maximum is 8.

By implication, 0 indicates no MLTA support is to be generated into the CCP and thus no $EMLD control statements are allowed. If this value is not 0, at least one $EMLD statement is required.

XLATE- { YES / NO }

The XLATE operand specifies whether programs always require translation of line transmission code to EBCDIC on input and translation from EBCDIC to line transmission code on output.

The default is YES indicating that translation is always used in MLTA communications operations. NO indicates that translation can sometimes be suppressed, via a specification in the assignment stage TERMATTR statement.

Regardless of the value of this operand, the forcing of upper case translation of input data is always allowed in the TERMATTR statement.

The value NO requires more CCP resident support than the value YES.

Note: That portion of the CCP that contains the MLTA IOCS and buffers ($CC4#2) must be loaded below 128K.

$EMLD–MLTA Devices

What You Must Know

_____ MLTA terminal devices to be supported.

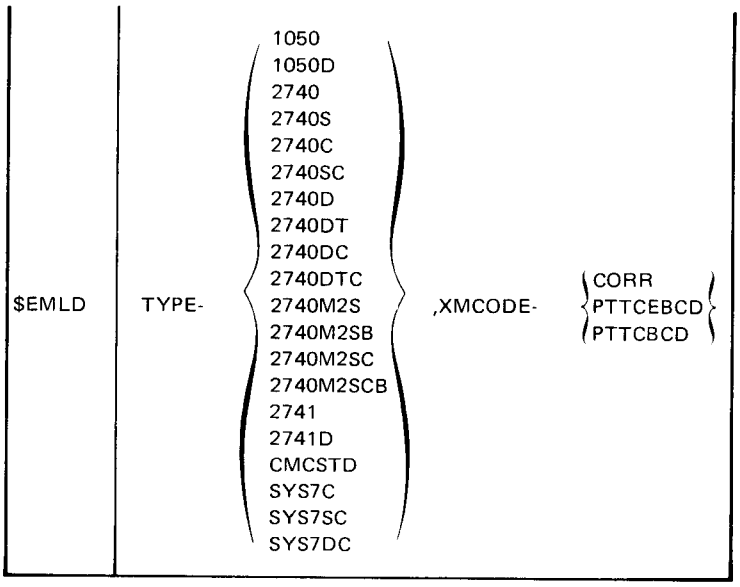
_____ MLTA line transmission code for each terminal type.

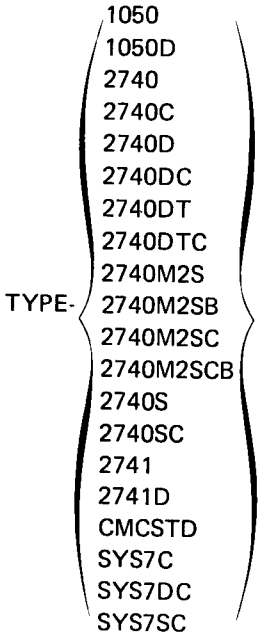
The $EMLD generation control statement indicates an MLTA terminal type to be supported with its features, the type of line the terminal is on, and the transmission code required on the line.

This statement is optional. It need be included only if the MLTA is supported.

If the MLTA is supported:

- At least one $EMLD statement is required.
- One $EMLD statement is required for each unique terminal-type/transmission-code combination (not one for each terminal).
- The first $EMLD statement must immediately follow a $EMLA statement. Other $EMLD statements must follow the first.
- Both operands on this statement are required.





The TYPE operand indicates the type of terminal and its features on a switched (dial) line or nonswitched (nondial) line.

The specification of a particular terminal type in a $EMLD statement causes the control logic support for that terminal type to be included in the CCP. Omission of a terminal type among the user's $EMLD statements indicates support for that type is not desired.

The following table explains the meaning of each terminal type code:

Terminal Type	Description
1050	1050 on a nonswitched line.
1050D	1050 on a dial (switched) line.
2740	2740 Model 1 without extra features on a nonswitched line.
2740C	2740 Model 1 with longitudinal redundancy checking (LRC) feature on a nonswitched line.
2740D	2740 Model 1 without extra features on a dial (switched) line.
2740DC	2740 Model 1 with longitudinal redundancy checking (LRC) feature on a dial (switched) line.
2740DT	2740 Model 1 with transmit control feature on a dial (switched) line.
2740DTC	2740 Model 1 with transmit control and longitudinal redundancy checking (LRC) features on a dial (switched) line.
2740M2S	2740 Model 2 with station control feature on a nonswitched line.
2740M2SB	2740 Model 2 with station control and buffer receive features on a nonswitched line.
2740M2SC	2740 Model 2 with station control and longitudinal redundancy checking (LRC) features on a nonswitched line.
2740M2SCB	2740 Model 2 with station control, longitudinal redundancy checking (LRC), and buffer receive features on a nonswitched line.
2740S	2740 Model 1 with station control feature on a nonswitched line.
2740SC	2740 Model 1 with station control and longitudinal redundancy checking (LRC) features on a nonswitched line.

Terminal Type	Description
2741	2741 on a nonswitched line.
2741D	2741 on a dial (switched) line.
CMCSTD	Communicating Magnetic Card SELECTRIC ® Typewriter on a dial (switched) line. The CMCST is supported to the extent that it functions identically to a 2741D.
SYS7C	System/7 functioning as a 2740 with longitudinal redundancy checking (LCR) feature on a nonswitched line. The System/7 is supported to the extent that it functions identically to a 2740C.
SYS7DC	System/7 functioning as a 2740 with longitudinal redundancy checking (LRC) feature on a dial (switched) line. The System/7 is supported to the extent that it functions identically to a 2740DC.
SYS7SC	System/7 functioning as a 2740 with station control and longitudinal redundancy checking (LRC) features on a nonswitched line. The System/7 is supported to the extent that it functions identically to a 2740SC.

The 3767 terminal can be used under CCP to simulate the following MLTA terminals: 2740C, 2740DC, 2740M2SCB, 2740SC, 2741, 2741D.

The only restriction on generation is that the CORR and PTTCEBCD operands are the only valid transmission codes when the 3767 is used to simulate one of the terminals listed above. (For assignment, see the MLTATERM statement under *Assignment Stage*.)

The 5100 portable computer can be used under CCP to simulate the following MLTA terminals: 2741 and 2741D. The only restriction on generation is that the CORR and PTTCEBCD operands are the only valid transmission codes when the 5100 is used to simulate one of the terminals listed above. (For assignment, see the MLTATERM statement under *Assignment Stage*.)

XMCODE- {CORR / PTTCEBCD / PTTCBCD}

The XMCODE operand indicates the MLTA line transmission code that the terminal in this $EMLD statement will use.

CORR indicates correspondence code

PTTCEBCD indicates paper tape transmission code EBCDIC

PTTCBCD indicates paper tape transmission code BCD (binary coded decimal)

Specifying a particular code causes generation to include the line-code-to-EBCDIC (upper and lower case) and EBCDIC-to-line-code translation modules when copying modules to the production pack.

The following table indicates the valid values of XMCODE for each terminal type and its features:

Terminal	Valid Transmission Code(s)
1050 1050D	PTTCEBCD
2740 2740C 2740D 2740DC 2740DT 2740DTC 2740M2S 2740M2SB 2740M2SC 2740M2SCB 2740S 2740SC	CORR PTTCEBCD PTTCBCD
2741 2741D	CORR PTTCEBCD PTTCBCD
CMCSTD	CORR
SYS7C SYS7DC SYS7SC	PTTCEBCD

$EBSC–BSC Support

What You Must Know

_____ BSCA lines, line features, and BSC control logic to be included in the CCP support.

_____ BSCA line transmission code for each terminal type.

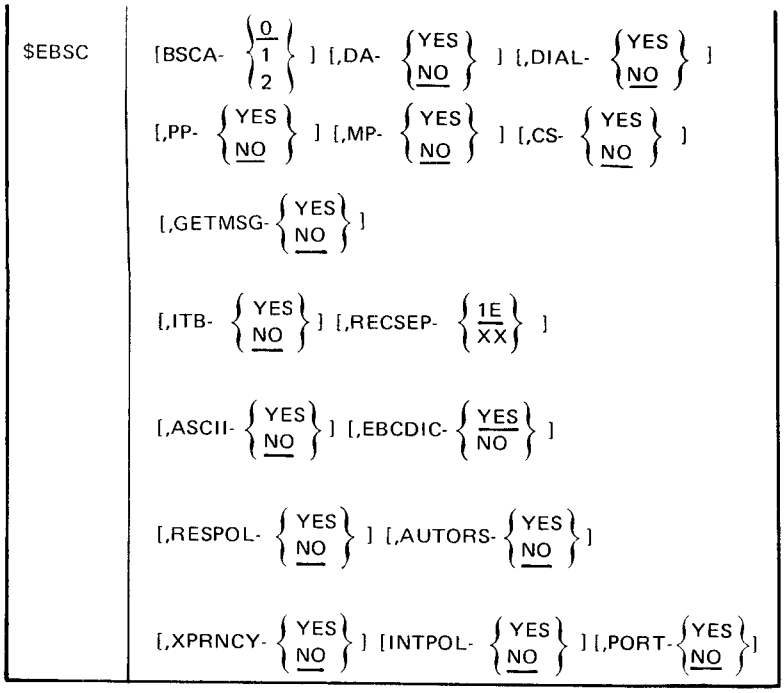
_____ Is the display adapter (DA) supported?

_____ Is internal polling supported?

The $EBSC generation control statement indicates general specifications concerning binary synchronous communications (BSC) support.

This statement is optional. It need be included only if the Binary Synchronous Communications Adapter (BSCA) is supported. The BSCA statements ($EBSC and $EBSD), the MLTA statements ($EMLA and $EMLD), or (for Program Number 5704-SC2 only) the BSCC statements ($ECSC and $ECSD) must be included. Both MLTA and BSCA can be specified. However, BSCC and MLTA are mutually exclusive. Only one $EBSC statement can be entered. If included:

- This $EBSC statement follows the last $EMLD statement if MLTA support is included. If MLTA support is not included, this statement immediately follows the $EFIL statement.
- At least one $EBSD statement is required to immediately follow this statement if the value of the BSCA operand is other than zero.



BSCA-{0 1 2}

The BSCA operand indicates the number of BSC adapters (lines) the CCP supports. The minimum (and default) is 0; the maximum is 2.

A value of 0 indicates that BSC support is not included, in which case, no other operands should be specified, (if specified, they must not have other than default values). Also, no $EBSD statements are allowed.

A value of 1 or 2 indicates that BSC control logic support is included in the CCP.

DA- {YES NO}

The DA operand indicates whether the display adapter is supported by CCP. DA-YES is valid only if the value of operand BSCA was 1 or 2. The default is NO. (See *Support of the Display Adapter* at the end of this chapter.)

DIAL- {YES NO}

The DIAL operand indicates whether a BSCA dial (switched line) network is to be supported by the CCP. The default is NO.

DIAL-YES is valid only if the value of operand BSCA was 1 or 2, and the 3275, 3735, 3741, CPU or processing units are supported.

YES includes the support and NO excludes it.

PP-{YES / NO}

The PP operand indicates whether a BSCA point-to-point (leased or private) network between a 3741 or processing units is supported by the CCP.

PP-YES is valid only if BSCA-1 or BSCA-2 was specified.

YES includes the support and NO excludes the support.

MP-{YES / NO}

The MP operand indicates whether a BSCA multipoint tributary (leased or private) network is supported under the CCP. This means the System/3 with the CCP is polled and addressed by another computer.

MP-YES is valid only if BSCA-1 or BSCA-2 was specified. YES includes the support and NO excludes the support.

The default is NO.

CS-{YES / NO}

The CS operand indicates whether a BSCA control station (leased or private) network is supported under the CCP. This means the System/3 with the CCP polls and addresses other terminals.

CS-YES is valid only if BSCA-1 or BSCA-2 was specified. YES includes the support and NO excludes the support.

CS-YES must be specified when using a 3271 control unit to control display terminals or when using a 3275 display station with nonswitched lines.

The default is NO.

GETMSG-{YES / NO}

The GETMSG operand indicates whether the BSCA control logic to *get a message* should be included in the CCP (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579).

YES includes the functional capability to read (in one user requested operation) from start-of-message to EOT. For example, one could read from many separated fields on the display screen of the 3277 and receive all the data as one continuous string. GETMSG-YES is required if FORMAT-YES is specified on the $EFAC statement or if program request under format is used. (See the PRUFLNG parameter of the PROGRAM statement in the assignment stage.)

The default is NO and excludes the support.

ITB-{YES / NO}

The ITB operand indicates whether BSC intermediate-text-block characters will be used in communicating with BSC terminals.

YES on this operand is valid only if BSCA-1 or BSCA-2 was specified. YES includes the control logic to handle fixed length block records with intermediate text-block characters. This control logic is used primarily for batch transmission devices, such as CPUs.

The default is NO and excludes the support.

RECSEP-{1E / xx}

The RECSEP operand specifies the hexadecimal record separator byte for BSC transmission. This feature is used primarily for batch transmission devices, such as CPUs, 3741, or 3735 terminals.

The user can specify a record separator byte as any two hexadecimal digits. When separator bytes are used, the most commonly used byte is hexadecimal 1E which is the default value. The record separator byte specified here is the character used in every instance of variable length block records.

ASCII-{YES / NO}

The ASCII operand specifies whether BSCA line 1 and/or line 2 uses ASCII transmission code.

YES includes the capability of translating between ASCII transmission code and EBCDIC code used by programs in the system. If BSCA is not 0, either ASCII-YES must be specified, or EBCDIC-NO must not be specified.

The default is NO and excludes the translation for ASCII code.

EBCDIC-{YES / NO}

The EBCDIC operand specifies whether BSCC line 1 and/or line 2 uses EBCDIC transmission code.

When no ASCII or EBCDIC operand is specified and BSCA is not 0, EBCDIC-YES is assumed.

The default is YES.

RESPOL-{YES / NO}

The RESPOL operand specifies whether BSC polling modules are main storage resident rather than executed as transients.

YES makes the polling routines resident. NO specifies polling is done by transients. There is no difference in function in the two methods. Main storage resident polling is used only for improved response time. The user can use this feature (main storage resident polling) when his programs can be interactive with many terminals on one line.

YES is valid only if BSCA-1 or BSCA-2 was specified in this same control statement. If RESPOL-YES is specified, CS-YES must be in effect.

Specifying NO or allowing this operand to default may cause performance to degenerate in the other program partitions due to disk reads of the polling transients.

The default is NO.

AUTORS-{YES / NO}

The AUTORS operand specifies whether the BSC multipoint tributary (non control station) support to automatically send a negative response to polling and addressing sequences is included in the CCP. This will be in effect only after the first data transfer has completed and will remain in effect until a subsequent BSCA I/O operation or a MLMP CLOSE operation is issued to the line. This allows the CCP to respond faster and avoid possible timeout problems on the CPU that is polling or addressing your system.

YES causes the proper module to reside in main storage. NO means that no response is given to the POLL/ADDRESS request from the control station until the program is loaded under CCP. Thus timeouts can occur at the control station. The user should specify YES if he expects serious timeout problems.

If AUTORS-YES is specified, MP-YES must be in effect. TYPE-CPU must be specified on a $EBSD generation control statement. The default for AUTORS is NO.

XPRNCY-{YES / NO}

The XPRNCY operand specifies whether the text transparency feature is used by application programs run under control of the CCP.

YES includes support for text-transparency. NO excludes the support.

If YES is specified, EBCDIC-NO must not be specified in this $EBSC statement.

The default is NO.

INTPOL-{YES / NO}

This keyword specifies whether to include support for timer-initiated interval polling. The default is NO. Interval polling allows the processing unit clock to stop between requests to poll BSC terminals during periods of low activity. All terminals are polled continuously until all terminals on the line have responded negatively for the time specified in the POLTIME parameter at the assignment stage. When a positive response is received from any BSC terminal, interval polling is suspended and continuous polling is resumed until another specified period of only negative responses.

Note: Interval polling is not recommended on a system with MINRES-YES (keyword $EGEN). It is not recommended because the loading of the additional non resident system routines causes the processing unit clock to run continuously.

PORT-{YES / NO} (5704-SC2 only)

The PORT operand indicates whether to include the facility. YES includes the logic to handle programs written to communicate over BSCA lines. Program Number 5799-WNK (Channel-Connected Systems RPQ) is not needed to support this facility. The default is NO and excludes the support.

What You Must Know

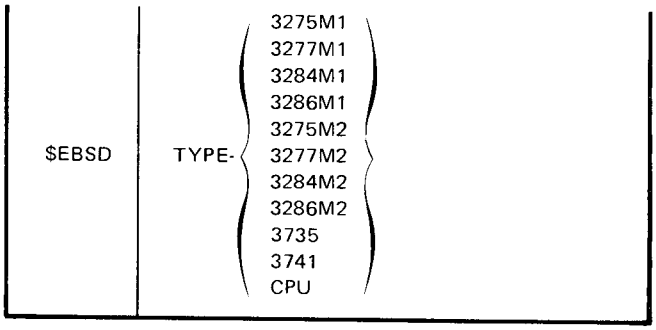
_____ BSCA terminal devices to be supported.

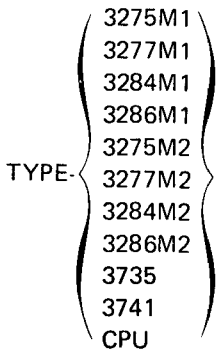
The $EBSD generation control statement indicates a BSC device type that the CCP is to support.

This statement is optional. It need be included only if BSC is to be supported.

If BSC is to be supported:

- At least one $EBSD statement is required.
- One $EBSD statement is required for each unique terminal type.
- The first $EBSD statement must immediately follow a $EBSC statement. Other $EBSD statements immediately follow the first. Multiple $EBSD statements may be in any order.
- The 3275M1 and 3275M2 require either control station or switched line support (CS-YES or DIAL-YES on the $EBSC statement). Other 3270 type terminals require control station line support (CS-YES on the $EBSC statement).
- The operand on this statement is required.





The TYPE operand specifies a particular BSC terminal type to be supported. Specification of a terminal type causes the necessary control logic to be included into the CCP to support that type. On individual $EBSD statements, any or all of the above terminal types can be specified.

The 3272 is not supported by the CCP.

M1 indicates a 480-byte buffer terminal and M2 indicates a 1920-byte buffer terminal of the 3270 system series.

3735 indicates the 3735 programmable terminal. For a description on how to create and transmit forms descriptor program (FDP), see *IBM 3735 Programmer's Guide*, GC30-3001; and *IBM 3735 Support Program Coding Manual*, GC21-5096.

CPU indicates support for all CPUs capable of receiving or transmitting in BSC (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579).

3741 indicates the 3741 Data Station Models 2 and 4 (see *IBM 3741 Data Station Reference Manual*, GA21-9183).

Note: The following terminals are supported in compatibility mode. Specify the appropriate entry for the terminal type as follows:

Terminal Type	Entry
3276	3277M2
3278	3277M2
3287	3284M2 or 3286M2
3288	3286M2
3289	3286M2
5231-2	3741

$ECSC—BSCC Support (5704-SC2 only)

What You Must Know

_____ BSCC lines, line features, and BSCC control logic to be included in the CCP support.

_____ BSCC line transmission code for each terminal type.

The $ECSC generation control statement indicates general specifications concerning binary synchronous communications controller (BSCC) support.

This statement is optional. It need be included only if the binary synchronous communications controller (BSCC) is supported. BSCC and MLTA are mutually exclusive within a single generation. Only one $ECSC statement can be entered. If included:

- The $ECSC statement follows the last $EBSD statement if BSCA support is included. If BSCA support is not included, this statement immediately follows the $EFIL statement.
- At least one $ECSD statement is required to immediately follow this statement if the value of the BSCC operand is other than zero.

$ECSC	[BSCC- {0 / 1 / 2}]
	[,GETMSG- {YES / NO}]
	[,ITB- {YES / NO}] [,RECSEP- {1E / XX}]
	[,ASCII- {YES / NO}] [,EBCDIC- {YES / NO}]
	[,XPRNCY- {YES / NO}] [INTPOL- {YES / NO}]
	[,CS- {YES / NO}][1] [,PP- {YES / NO}][1]
	[,PORT- {YES / NO}][1]

BSCC- {0 / 1 / 2}

The BSCC operand indicates the number of BSCC adapters (lines) that CCP supports. The minimum (and the default) is 0; the maximum is 2.

A value of 0 indicates that BSCC support is not included. In this case, no other operand should be specified (if they are specified, they must not have other than the default value). Also, no $ECSD statements are allowed.

A value of 1 or 2 indicates that BSCC control logic support is included in CCP.

GETMSG- {YES / NO}

The GETMSG operand indicates whether the BSCC control logic to *get a message* should be included in the CCP (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579).

YES includes the functional capability to read (in one user requested operation) from start-of-message to EOT. For example, one could read from many separated fields on the display screen of the 3277 and receive all the data as one continuous string. GETMSG-YES is required if FORMAT-YES is specified on the $EFAC statement or if program request under format is used. (See the PRUFLNG parameter of the PROGRAM statement in the assignment stage.)

The default is NO and excludes the support.

ITB- {YES / NO}

The ITB operand indicates whether BSCC intermediate-text-block characters will be used in communicating with BSCC terminals. ITB-YES cannot be specified if XPRNCY-YES is specified.

YES on this operand is valid only if BSCC-1 or BSCC-2 was specified. YES includes the control logic to handle fixed length block records with intermediate text-block characters. This control logic is used primarily for batch transmission devices, such as CPUs.

The default is NO and excludes the support.

[1] Program Number 5704-SC2 only

RECSEP- {1E / xx}

The RECSEP operand specifies the hexadecimal record separator byte for BSCC transmission. A CCP generated to support BSCC terminals always includes code for handling variable length records. This code is used primarily for batch transmission devices, such as CPUs, 3741, or 3735 terminals.

The user can specify a record separator byte as any two hexadecimal digits. When separator bytes are used, the most commonly used byte is hexadecimal 1E which is the default value. The record separator byte specified here is the character used in every instance of variable length block records.

ASCII- {YES / NO}

The ASCII operand specifies whether any CCP supported BSCC line uses ASCII transmission code.

YES includes the capability of translating between ASCII transmission code and EBCDIC code used by programs in the system. If BSCC is not 0, either ASCII-YES must be specified, or EBCDIC-NO must not be specified.

The default is NO and excludes the translation for ASCII code.

EBCDIC- {YES / NO}

The EBCDIC operand specifies whether any CCP supported BSCC line uses EBCDIC transmission code.

When no ASCII or EBCDIC operand is specified and BSCC is not 0, EBCDIC-YES is assumed.

The default is YES.

XPRNCY- {YES / NO}

The XPRNCY operand specifies whether the text transparency feature is used by application programs run under control of the CCP.

YES includes support for text-transparency. NO excludes the support.

If YES is specified, EBCDIC-NO must not be specified in this $ECSC statement.

The default is NO.

INTPOL- {YES / NO}

This keyword specifies whether to include support for timer-initiated interval polling. The default is NO. Interval polling allows the processing unit clock to stop between requests to poll BSCC terminals during periods of low activity. All terminals are polled continuously until all terminals on the line have responded negatively for the time specified in the CSPOLTIM parameter at the assignment stage. When a positive response is received from any BSCC terminal, interval polling is suspended and continuous polling is resumed until another specified period of only negative responses.

PP- {YES / NO} (5704-SC2 only)

The PP operand indicates whether a BSCC point-to-point (leased or private) network between a System/3 and a processing unit (or 3741) is supported by CCP. PP-YES is valid only if BSCC-1 or BSCC-2 was specified. YES includes the support and NO excludes the support. The default is NO.

CS- {YES / NO} (5704-SC2 only)

The CS operand indicates whether a BSCC control station network is supported under CCP. This means that the System/3 with CCP polls and addresses terminals. CS-YES is valid only if BSCC-1 or BSCC-2 was specified. CS-YES must be specified when using a 3271 control unit to control display terminals or when using a 3275 display station with nonswitched lines. YES includes the support and NO excludes the support. The default is YES.

PORT- {YES / NO} (5704-SC2 only)

The PORT operand indicates whether BSCA portline facility is to be supported by CCP. YES includes the logic to handle programs written to communicate over SIOC lines. Program Number 5799-WNK (Channel-Connected Systems RPQ) is not needed to support this facility. The default is NO and excludes the support.

What You Must Know

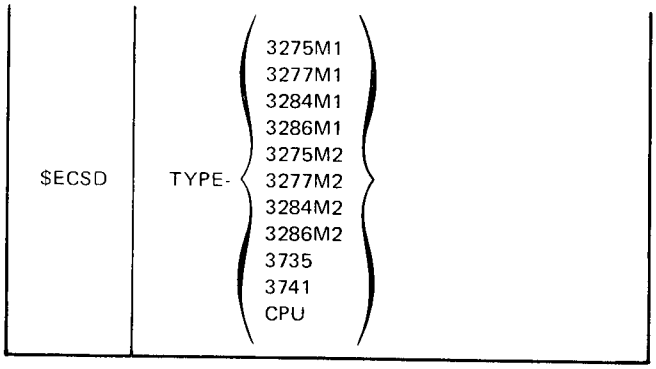
_____ BSCC terminal devices to be supported.

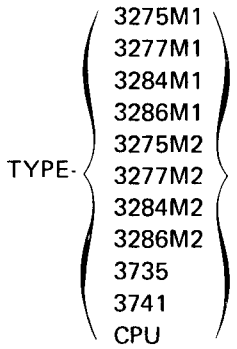
The $ECSD generation control statement indicates a BSCC device type that the CCP is to support.

This statement is optional. It need be included only if BSCC is to be supported.

If BSCC is to be supported:

- At least one $ECSD statement is required.
- One $ECSD statement is required for each unique terminal type.
- The first $ECSD statement must immediately follow a $ECSC statement. Other $ECSD statements immediately follow the first. Multiple $ECSD statements may be in any order.
- The operand on this statement is required.





The TYPE operand specifies a particular BSCC terminal type to be supported. Specification of a terminal type causes the necessary control logic to be included into the CCP to support that type. On individual $ECSD statements, any or all of the above terminal types can be specified.

The 3272 is not supported by the CCP.

M1 indicates a 480-byte buffer terminal and M2 indicates a 1920-byte buffer terminal of the 3270 system series.

3735 indicates the 3735 programmable terminal. For a description on how to create and transmit forms descriptor program (FDP), see *IBM 3735 Programmer's Guide*, GC30-3001; and *IBM 3735 Support Program Coding Manual*, GC21-5096.

CPU indicates support for all CPUs capable of receiving or transmitting in BSCC (see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579).

3741 indicates the 3741 Data Station Models 2 and 4 (see *IBM 3741 Data Station Reference Manual*, GA21-9183).

Note: The following terminals are supported in compatibility mode. Specify the appropriate entry for the terminal type as follows:

Terminal Type	Entry
3276	3277M2
3278	3277M2
3287	3284M2 or 3286M2
3288	3286M2
3289	3286M2
5231-2	3741

$EGEN–CCP Generation Stream

What You Must Know

_____ Disk unit and pack name on which disk system management resides (F1 or R1).

_____ Disk unit and pack name on which the CCP is generated.

_____ Disk unit(s) and pack name(s) for work file space during generation.

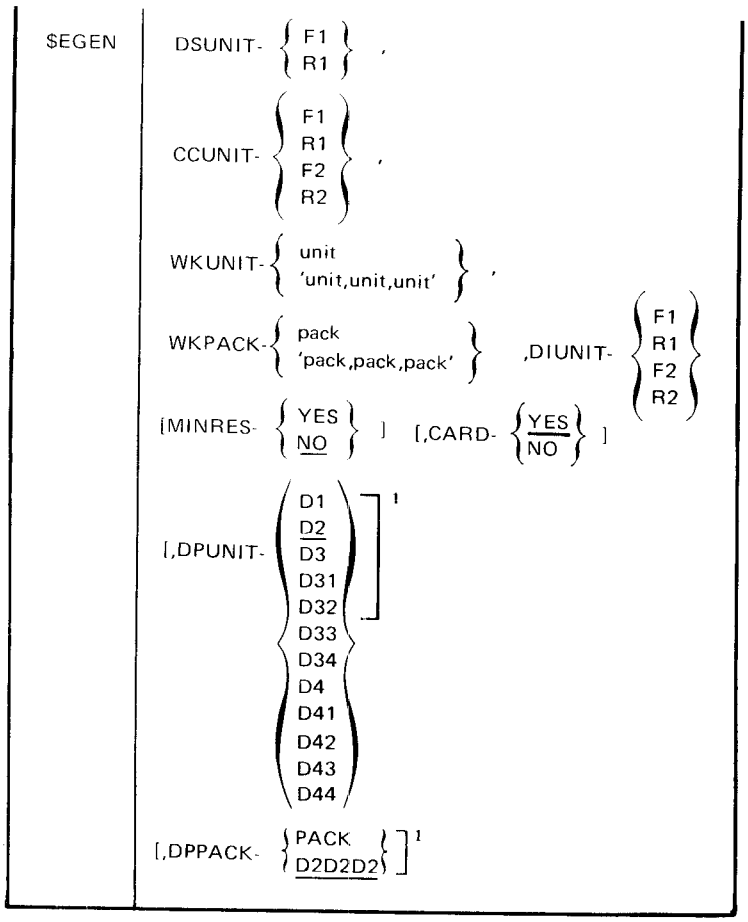
_____ Disk unit on which the macro processor and the distribution CCP modules reside.

The $EGEN generation control statement indicates where various unit and pack names are located during the CCP generation, permits the user to specify the minimum size of resident code, and determines the input medium used during step 6 of generation.

This statement is always required. It must be the last of all $E generation control statements. Only one $EGEN statement can be entered.

The disk units used in a CCP generation are not fixed, as in a generation of DSM. Any available unit can be used to hold any pack required, if:

- The distribution pack remains mounted on the same unit throughout generation.
- The system pack remains mounted on the same unit throughout generation.
- The distribution pack is not used as the receiving pack for any relocatable or load modules produced by generation.



[1] For Program Number 5704-SC2 only

DSUNIT- { F1 / R1 }

The DSUNIT operand specifies which disk unit (F1 or R1) contains the DSM during the CCP generation. This is the unit from which the user loads the DSM (performs IPL). It is assumed that the DSM is properly generated on the system pack, including the appropriate MLTA and/or BSCA I/O macros and subroutines. Either of the values is valid. There is no default.

CCUNIT- { F1 / R1 / F2 / R2 }

The CCUNIT operand specifies the production disk unit upon which the generated CCP modules, except for those which directly support application program preparation (specified in the $EPLG statement), are to be generated. Any of the values is valid. There is no default. This unit can be the same as that specified for DSUNIT.

WKUNIT- { unit / 'unit,unit,unit' }

The WKUNIT operand specifies the disk unit(s) upon which work files are to be allocated during the CCP generation. The value for unit can be R1, F1, R2, F2, D1, D2, D3, or D4; for Program Number 5704-SC2, D31, D32, D33, D34, D41, D42, D43, and D44 are also valid. There is no default value.

Either the single unit form or the triple unit form must be used. The same form must be used for the WKPACK operand. If the single unit form is used, $SOURCE, $WORK, and $WORK2 are allocated on the same (specified) unit whenever they are required.

If the triple unit form is used, the first unit indicates where $SOURCE is allocated whenever it is required. The second unit indicates where $WORK is allocated whenever it is required. The third unit indicates where $WORK2 is allocated whenever it is required.

The work space required is:

- For $SOURCE: 170 tracks
- For $WORK: 50 tracks
- For $WORK2: 50 tracks

The time required for generation can be significantly reduced if these files (particularly $SOURCE) can be allocated on a pack other than the CCP distribution pack, especially if that pack can be on a unit served by a separate disk arm.

WKPACK-{pack / 'pack,pack,pack'}

The WKPACK operand specifies the pack name(s) upon which the work files are allocated during the CCP generation. The value for pack is the appropriate user's pack name. There is no default value.

Either the single pack form or the triple pack form must be used. The same form must be used as for the WKUNIT operand. If the single unit form was used for WKUNIT, then the single pack form must be used here. If the triple unit form was used for WKUNIT, then the triple pack form must be used here. If the single pack form is used, $SOURCE, $WORK, and $WORK2 will all be allocated on the same (specified) pack whenever they are required.

If the triple pack form is used, the first pack indicates the pack name for allocation of $SOURCE whenever it is required. The second pack indicates the pack name for allocation of $WORK whenever it is required. The third pack indicates the pack name for allocation of $WORK2 whenever it is required.

DIUNIT-{F1 / R1 / F2 / R2}

The DIUNIT operand specifies the disk unit which contains the macro processor and the distribution CCP modules as shipped from the IBM Program Information Department (PID). The value must not specify a unit that is the same as that specified for CCUNIT, nor the same as any specified for PPUNIT in a $EPLG statement.

MINRES-{YES / NO}

The MINRES operand specifies whether the size of the resident CCP control routine is made as small as possible.

YES indicates that the resident control routine is made as small as possible. This is done by making certain communications handling routines transient rather than resident.

Note: Interval polling is not recommended on a system with MINRES-YES (keyword $EGEN). It is not recommended because the loading of the additional non-resident system routines causes the processing unit clock to run continuously.

CARD- {YES / NO}

The card operand specifies the input medium to be used during Step 6 of the CCP generation. YES, the default value, indicates a card-oriented generation. In this case, the complete job stream is punched during generation. After being punched, the job stream is entered via the card reader to complete the generation.

NO indicates a cardless-oriented generation.

DPUNIT (5704-SC2 only)[1]

This operand specifies the unit on which the CCP dump file ($CCPDUMP) resides. The default is D2 (5704-SC2 only).

DPPACK (5704-SC2 only)[1]

This operand specifies the pack on which the CCP dump file ($CCPDUMP) resides. The default is D2D2D2.

[1] When running a card generation, the punched cards for DPUNIT and DPPACK from the sample deck should be filled in. If the default is to be taken, the cards should be pulled from the sample deck and left out for step 3 of the generation.

CCP PROGRAMS USED IN GENERATION

Generation Utility ($CC1PP)

The generation utility program ($CC1PP) is used in step 5 of the CCP generation to print any error messages, or if there are no errors, to print and provide the input stream for the second pass of generation.

For a card-oriented generation, the output from $CC1PP can be punched on either the MFCM, MFCU, or the 1442, whichever is the standard punch device for the system on which generation occurred.

SCP Generator ($CGxxx)

The system control program generator ($CGDRV and related $CGxxx phases) is used in the second pass of generation (step 6) to generate:

- The initialization data for $CCPFILE.
- The execution time resident load modules ($CC4#1, $CC4#2, and, if BSCC is supported, $CC4#M).
- A null user security information module if a user security routine is used.
- A vector table, $CC4VT.

Initialize Assignment File Build ($CC1BF)

The initialize assignment file build program ($CC1BF) allocates and initializes the $CCPFILE file in step 8 of the CCP generation to prepare the $CCPFILE file for the assignment stage.

Reexecution of $CC1BF (after proper scratching of $CCPFILE) reinitializes $CCPFILE to the state it was before the first assignment set was entered.

By changing the unit and pack name on the // FILE card of $CC1BF, $CCPFILE can be initialized on another unit and pack. By changing the TRACKS parameter of the // FILE card, the size of the new $CCPFILE can be changed. Multiple $CCPFILEs can exist if each is on a separate pack. Selection of the unit for the appropriate $CCPFILE can be made at startup.

Initialize Trace/Dump File ($CC1DP) (Program Number 5704-SC2 only)

The initialize CCP dump file program ($CC1DP) allocates and initializes the $CCPDUMP file for trace and storage dumps. The $CCPDUMP file consists of a directory, a trace dump area, and a main storage dump area. The file may reside in any of the main data areas, and its location may be designated during either CCP generation by the specification of the DPPACK and DPUNIT operands of the $EGEN macro or CCP assignment by the specification of the DUMPUNIT operand on the // SYSTEM statement. The track space desired for the trace area may be specified during generation by the DPTRAC operand on the $EFIL statement. During the generation stage, you can calculate the necessary track space for storage dumps by analyzing the CORE and DUMP operands of the $EFIL statement.

CCP generation creates and executes the proper procedure for $CCPDUMP initialization. If the user wishes to generate his own procedure, he can use the following example:

```
// LOAD $CC1DP,R1
// FILE NAME-$CCPDUMP,UNIT-D2,PACK-D2D2D2,
   RETAIN-P,TRACKS-97
// RUN
// TRACE TRACKS-10
/*
```

The // TRACE statement identifies to $CCP1DP the number of tracks desired for trace. The remaining 86 tracks are allocated for main storage dumps. The 86 track calculation is as follows:

$$\left[\left((512 \times 4) +1\right) \times 2\right]/48 = 86 \left\{\begin{matrix} \text{512K system,} \\ \text{2 dumps needed} \end{matrix}\right\}$$

$$\left[\left((\text{main storage in K} \times 4) +1\right) \times \text{number of dumps}\right]/48$$

Note: In addition to the total needed for trace and storage dumps, the user should specify one track on his // FILE OCL statement; this specification allocates the necessary space for $CCPDUMP directory information.

Note: To change the size of a $CCPDUMP file, delete it and recreate it using $CC1DP as outlined above.

SOURCE MODULES USED IN GENERATION

In addition to the source library modules used for generating the execution time resident code or supporting transients, the following source library members are present on the distribution pack:

$CGSST	Sample assignment control statements – cardless
$CG1G1	Cardless CCP generation instructions
$CG1GG	Sample $E macro statements – cardless
$CG1G1	Cardless CCP generation procedure
$CG1G3	Cardless CCP generation procedure
$CG1G4	Cardless CCP generation procedure
$CG1G5	Cardless CCP generation procedure
$CGSMP	CCP sample generation deck
$CGEND	Instructions to the user at the end of generation
$CGSET	Sample assignment input deck and sample startup deck for the installation verification program (CCPIVP)

GENERATION CONSIDERATIONS

One of three versions of BSC support for CCP can be selected during the CCP generation process:

- Full BSC support for all line configurations and BSC devices.
- 3270 only support for all line configurations.
- Privileged 3270 only support for faster response.

If other than 3270 devices are specified by the $EBSD statements, full BSC support is generated. If only 3270 devices are specified by the $EBSD statements, two versions of BSC IOS can be supported:

- 3270 only support, which saves approximately 1K bytes.
- Privileged 3270 only support, which saves approximately .75K bytes and gains faster response times during heavy usage. Privileged 3270 only support is recognized only if MINRES-NO ($EGEN statement), RESPOL-YES ($EBSC statement), and 3270 only devices ($EBSD statement) are requested.

Support of the Display Adapter

Support of the Display Adapter on your system requires special CCP generation and assignment considerations. The following are the generation and assignment statements that have required values for a system that supports the display adapter only (PRUF and DFF included). The underlined parameters are required values for the Display Adapter. If a parameter is not given or not underlined, it is not specifically required nor has any effect on the display adapter support. Some of the parameters default to the value shown (for example, MP-NO).

Generation

$EBSC BSCA-1[1],DA-YES,DIAL-NO,PP-NO,MP-NO, CS-YES,GETMSG-YES,ITB-NO, EBCDIC-YES,AUTORS-NO,INTPOL-NO

$EBSD TYPE-3277Mx[2]

Assignment

// SYSTEM POLTIME-NO, DFFPACK-{SYSTEM / PROGRAM}

// TERMATTR TRANSLAT-NO,SWITCHED-NO, BLKL-512,DATAFORM-MESSAGE, ITB-NO,VARL-NO,SPAN-NO, DFF3270-YES

// BSCALINE TYPE-CS,LINENUM-2, XMCODE-EBCDIC,POLLIST-termid [,termid] . . .

// BSCATERM TERMID-termid,TYPE-3277Mx[2], COMMAND-{NO / YES}, ONLINE-{YES / NO}, OFFACTN-{HOLD / DROP}, ADDRCHAR-6060xxxx[3], POLLCHAR-4040xxxx[3]

// TERMNAME NAME-termname,TERMID-termid

// PROGRAM NAME-pgmname,MRTMAX-n, PGMDATA-YES,PRUF$Z-formatname, PRUFLNG-nnnn,DFFMTERM-n, DFFNDF-n,DFFSFDT-n

[1] BSCA-2 is also valid.
[2] 3284Mx and 3286Mx are also valid (x = Model number).
[3] xxxx equals the device address.

Support of BSCC (Binary Synchronous Communications Controller)

BSCC is the same as BSC lines 3 and/or 4. This documentation refers to BSCC communications scheduler and to $CC4#M (BSCC resident code). These two terms are interchangeable.

To provide maximum flexibility, CCP generation has two macros, unique to BSCC, that define line and device support.

Macro $ECSC (BSCC line support) and $ECSD (BSCC device support) produce the resident BSCC module $CC4#M when used during CCP generation. $CC4#M serves a function for BSCC similar to the function that $CC4#2 serves for BSCA.

// BSCALINE and // BSCATERM (5704-SC2) assignment statements cause the BSCC interface to CCP assignment stage to be implemented. BSCC lines may be either control station or point-to-point. Outboard polling may be included (BSCC only) by specifying OUTPOLL-YES and TYPE-CS (or allowing the TYPE to default to YES) in the // BSCALINE statement. The other parameters are processed the same as they would have been if LINENUM-{1 / 2} had been specified.

Support of a Large Number of Terminals (Program Number 5704-SC2 only)

Because of addressing considerations within the resident portion of CCP, there are limitations placed on the number of terminals that can be supported by CCP. This limitation is affected by the number of programs, the size of line buffers, whether DFF is supported, and so on. In order to increase the number of terminals that may be supported and to free some additional space in the resident area, two generation options may be selected; these options will move portions of the resident CCP to the user program area (UPA). If more than 50-60 terminals are to be supported in any one assignment set, the following options (both on the EFAC statement) should be considered:

[,MOVDFF-{YES / NO}]

[,MOVTNT-{YES / NO}]

Specifying MOVDFF-YES will move the DFF control routine to the UPA, thereby freeing more than 3K bytes of resident area and occupying 4K bytes of the UPA.

Specifying MOVTNT-YES will cause the Terminal Name Table to be built in the UPA, using either 2K or 4K bytes of UPA. A 2K Terminal Name Table (TNT) will support 184 terminal names. If more than 184 terminal names are specified at assignment time, 4K bytes of UPA will be used.

During the CCP generation stage, the user fixes the maximum size and capabilities of the CCP. During the assignment stage of the CCP, he describes the CCP operating environment in more detail.

During assignment, the user defines one or more *sets* of terminals, files, programs, and system environments that are available to the CCP. These assignment sets are recorded in the assignment file ($CCPFILE) allocated during the generation stage (see Chapter 6. *Generation Stage*). Each time the CCP is run, it operates under one of the assignment sets; that is, the CCP has access to a particular group of terminals, files, and programs. The user can restrict the resources defined by a set during operation startup (see index entry: *operational startup*). This allows him, for example, to control which programs are eligible to be called during a particular CCP run or to restrict certain files during a run.

While the range of function specified during the CCP generation will not vary by assignment set, the specific operating environment of the CCP execution might vary.

The information placed in an assignment set during the assignment stage is valid until a terminal, program, or file must be added to or removed from the set, or until aspects of the system environment change. When the system environment changes, the contents of the assignment file can be modified by repeating the assignment run, without regenerating the CCP.

PLANNING FOR ASSIGNMENT

The assignment run must be repeated each time a new program or file is added to a set, each time the group of terminals available to the CCP changes, or each time certain other aspects of the CCP configuration change. The assignment stage of the CCP can be run frequently or infrequently, depending upon how often the CCP environment changes.

If the CCP is to run under various sets of assignments in a given period of time, more than one set of assignments can be placed in the assignment file. For example, a certain set of terminals, files, and programs can be available during the day, with a restricted set available during night-time hours. Or, perhaps, a certain group of files, terminals, and programs are required for the weekend, month-end, or year-end operations.

Information from the assignment file, $CCPFILE, is required by various people. The system manager and the system operator must be aware of all current system assignments so they can properly control and maintain the communication-based system. The terminal operator should be aware of some aspects of the system assignments, including which symbolic names are assigned to his terminal and what the current password or other sign-on security information is. Programmers also require current information about the system assignments, such as the symbolic file names and the actual files they refer to. In order to make this information available to those who need it, provisions must be made to distribute all or part of the information from the assignment file whenever it changes.

REQUIREMENTS FOR ASSIGNMENT

In order to execute the assignment stage of the CCP, the system must meet minimum requirements. These requirements are less than are needed for the CCP operational stage. The requirements are:

- 22K (Program Number 5704-SC1), or 36K (Program Number 5704-SC2) of main storage available for the CCP assignment stage to execute.
- One 5444 Model 2 disk storage drive, or one 3340 (or 3340 simulation area).
- An input device.
- A 1403 Printer.

The assignment build program, $CCPAS, and the assignment list program, $CCPAL, operate under control of the System/3 Model 15 disk system management. The user must provide two OCL file statements when loading the assignment build program, one for the assignment file, $CCPFILE, and one for a work file ($CCPWORK).

ASSIGNMENT RESTRICTIONS (5704-SC2 ONLY)

The assignment build program ($CCPAS) or assignment list program ($CCPAL) can be executed after the $CCPFILE build program ($CC1BF) has been run. Either $CCPAS or $CCPAL may be executed while CCP is in the execution phase in another partition. The executing assignment set can be modified while CCP is executing in another partition. However, the modification will not be observed until CCP is shut down and then restarted using the modified assignment set. There are, however, a few restrictions involved in running either $CCPAS or $CCPAL. They are as follows:

- $CCPAS will only perform diagnostics and will not update $CCPFILE if any of the following conditions occur:
  - $CCP is in startup phase in another partition.
  - $CCP is in shutdown phase (or shutdown is pending) in another partition using the same $CCPFILE that $CCPAS is trying to alter.
  - $CCPAL is executing in another partition.
  - Another copy of $CCPAS is executing in another partition.
  - CCPFMT or CCPPGM is updating the same $CCPFILE that $CCPAS is trying to alter.

- $CCPAL will not start executing if any of the following conditions occur:
  - $CCP is in startup phase in another partition.
  - $CCP is in shutdown phase (or shutdown is pending in another partition using the same $CCPFILE that $CCPAL is trying to list.
  - $CCPAS is executing in another partition.
  - Another copy of $CCPAL is executing in another partition.
  - CCPFMT or CCPPGM is updating the same $CCPFILE that $CCPAL is trying to list.

ASSIGNMENT DIAGNOSTICS

The CCP assignment stage, $CCPAS, analyzes each of the assignment statements for invalid specifications. If an error is found in any statement of a set, that set of statements is not processed by the assignment stage; however, the remaining statements in that set are also analyzed for syntax errors.

The assignment list program, $CCPAL, analyzes the list statement for errors. Error messages are written to the system print device.

ASSIGNMENT FILE ($CCPFILE)

During an assignment run, the user defines a specific CCP run environment in a set of tables contained in the CCP assignment file on disk. This file, allocated and initialized during the CCP generation, always has the name, $CCPFILE. When the user starts operating his CCP, the startup routines of the CCP, using the information in $CCPFILE, initialize the CCP for this run so that only a specific set of terminals, files, and programs indicated by the user can be used.

A set of specifications in the assignment file, once set by the user in an assignment run, hold for any number of operational runs of the user's CCP. The assignment file need be changed only when one or more sets of specifications require change.

Assignment allows the user to have more than one operational environment to run the CCP. He might, for example, have one set of terminals, files, and programs to use during the day, and a different (though possible overlapping) set for night. His requirements might even be for more than two such environments. The assignment file may, if the user desires, contain several different sets of specifics.

The user can have more than one $CCPFILE but only one per pack. The CCP system operator can, upon any startup of an operational run of the CCP, identify the unit containing the appropriate $CCPFILE and the set of specifics that shall apply to the current run.

Note: The maximum number of sets allowed in one assignment file is 25.

ASSIGNMENT STATEMENTS (ASSIGNMENT BUILD PROGRAM)

In order to place an assignment set in the assignment file, $CCPFILE, the user must run the assignment build program, $CCPAS. The user's input to the assignment program consists of a series of statements, similar to OCL statements. These statements identify the programs, files, terminals, and certain system options that constitute a CCP operating environment. The assignment statements are printed by $CCPAS on the system print device.

The assignment build program analyzes the statements to ensure that they are valid. Following successful validation, the specifications are encoded by $CCPAS and written into the assignment file as an assignment set (for example, a group of tables in the assignment file, $CCPFILE, which defines one CCP operating environment). The assignment program can be used to create new assignment sets, delete existing assignment sets, or replace existing sets with new specifications. Different assignment sets can be created, deleted, and replaced in the same run of the assignment program.

A set of assignment statements begins with a statement identifying the assignment set and indicates whether it is to be modified, created, deleted, or replaced. Only this statement is required to delete a set. Subsequent statements provide the specific information to be placed in the assignment set. When an item in an assignment set is replaced, a complete set of assignment statements must be entered (except for the modification of certain control information).

$CCPAS may be run in any of four execution modes:

1. Create a new assignment set.
2. Delete an existing assignment set.
3. Replace an existing assignment set.
4. Modify the system environment portion of an existing assignment set.

The input for $CCPAS is as follows:

- The assignment file $CCPFILE.
- The OCL/FILE statements provided by the user: The following OCL statements are needed:

```
1   4       8       12      16      20      24      28      32
//  LOAD  $CCPAS,unit
//  FILE  NAME-$CCPFILE,UNIT-unit,
//        PACK-packid
//  FILE  NAME-$CCPWORK,UNIT-unit,
//        PACK-packid,RETAIN-S,
//        TRACKS-n
//  RUN
```

Notes:

1. The unit parameter for $CCPAS specifies the location of the assignment build program.
2. The unit parameter for $CCPFILE specifies the location of the assignment file to be used and must be a 5444 unit.
3. $CCPWORK should have RETAIN-S specified to allow deletion and can be on R1, F1, R2, or F2.
4. The keyword RECORDS should not be used to define the size of $CCPWORK; instead the keyword TRACKS should be used. Additionally, the size must be large enough to contain the largest set to be processed (see *Disk Storage Estimates for the CCP* in Appendix F.

- The assignment control statements instructing $CCPAS of the function(s) to perform. $CCPAS reads the control statements from the system input device and/or from a source library. Each group of control statements, grouped together by the user to define an operating environment, is called an *assignment set.* The input to $CCPAS can consist of one or more sets of control statements.

// SOURCE	This optional statement indicates that control statements are read from a source library member. One or more SOURCE statements can be included anywhere within a set of control statements.

The following control statements define an assignment set and must be specified in the order shown:

// SET	This required statement identifies the output assignment set operated on and the type of operation to be performed. If multiple input assignment sets are provided as input to $CCPAS, the // SET statement defines the beginning of each assignment set.
// SYSTEM	This required statement defines certain operational environment options not directly related to terminals, files, or programs when this set is used for execution of the CCP.

// TERMATTR

Each of these statements defines the variable attributes of a terminal. Each terminal statement (BSCATERM and MLTATERM) must refer to a TERMATTR statement.

Note: Any BSCALINE statement with corresponding BSCATERM statements (as a group) can appear before or after any MLTALINE/MLTATERM group. The order of input lines determines the priority of service during execution of the CCP. For example, the order might be:

MLTALINE
MLTATERM (one or more)
BSCALINE
BSCATERM (one or more)
MLTALINE
MLTATERM (one or more)

// BSCALINE

Each of these optional statements identifies a BSC line, its characteristics, and the order the terminals on that line (if there is more than one terminal) shall be polled.

// BSCATERM

At least one of these statements must follow a BSCALINE statement. Each BSCATERM statement identifies a terminal on the line, its operating characteristics, and whether it can issue commands to the CCP.

Note: The BSCATERM statement must follow the BSCALINE statement for which the terminal is defined.

// MLTALINE

Each of these optional statements identifies a MLTA line, similar to the BSCALINE statement

// MLTATERM

At least one of these statements must follow a MLTALINE statement. Each MLTATERM statement identifies a terminal on the line, its operating characteristics, and whether it can issue commands to the CCP.

Note: The MLTATERM statement must follow the MLTALINE statement for which the terminal is defined.

// PORTLINE (5704-SC2 only)

The PORTLINE statement defines how the BSCA, BSCC, or SIOC portline is to be divided into ports and the size of the line buffers. The PORTLINE statement should be entered after the SYSTEM statement or after the last TERMATTR statement. One PORTLINE statement may be entered for each BSCA, BSCC or SIOC line supported. For SIOC support, Program Number 5799-WNK is also required.

// TERMNAME

Each of these required statements defines a symbolic terminal name by which a terminal can be referred to in an application program. The symbolic terminal name can be assigned to a physical terminal or can be left unassigned.

// DISKFILE

Each of these optional statements defines a user disk data file and the physical attributes of the file.

// SYMFILE

Each of these optional statements defines a symbolic name that can be used to reference one or more disk files specified on // DISKFILE statements.

// PROGRAM

Each of these required statements identifies a user program that is permitted to execute during a CCP run using this set. The statement defines the program resource requirements and operational characteristics.

When creating or replacing an assignment set, at least one BSCALINE and one BSCATERM statement, or one MLTALINE and one MLTATERM statement must be in the control statements.

- A /* statement must be given at the end of all input control statements except when the last control statement is from the source library. In this case, see the keyword END in the SOURCE statement.

The output for the assignment build program is as follows:

- A listing of the assignment control statements on the printer.
- Appropriate diagnostics on the printer and an error halt, at the end of the run, if necessary.
- The $CCPFILE containing created, replaced, or modified assignment sets, and no longer containing any sets specified to be deleted. The $CCPFILE is ready for operational startup.

An assignment set will be created or replaced in the assignment file only if the complete set, as read by the assignment build program, is free of errors.

The $CCPAS cannot run concurrently in both partitions nor run concurrently with the CCP in the other partition. ($CCPFILE cannot be validly altered from both partitions concurrently or while the CCP is running.)

ASSIGNMENT CONTROL STATEMENTS

Each control statement contains an identifier and parameters. The identifier is a term that defines the type of control statement and is always the first field of the statement following the //. The parameters are the control information being supplied to the program. Each parameter consists of a keyword to identify the parameter, a hyphen, and the appropriate control information value. Parameters within any single control statement can be entered in any order.

Writing Assignment Control Statements

The rules for constructing assignment control statements are as contained in the following paragraphs.

Statement Identifier: // in columns 1 and 2 followed by at least one blank preceding the statement identifier.

Blanks: One or more blanks is required between the // and the statement identifier, and between the statement identifier and the first keyword. The first blank following a keyword value indicates the end of the statement. A blank following a comma (which follows a value given for a keyword parameter) indicates that the statement is continued on the following input record (which must have a //b in the first three columns).

Keyword Parameters: These parameters are separated by commas. A hyphen (-) separates each keyword from the corresponding value. Blanks are not allowed within or between parameters.

Sublist: A sublist is a series of values given for a single keyword. Each value is related (such as a series of filenames). The first value of a sublist always has an apostrophe (') entered before the value, succeeding values are separated by commas, and the last value has an apostrophe after it. For example: EXAMPLE-'value, value,value'.

Split-Value: A split-value is a group of values that might be given for a keyword parameter; however, each has a separate and distinct relationship to the keyword. These split-values are separated by a slash (/). If a series of split-values is given, the series is called a split-value sublist. If a split-value is not given as a sublist, then apostrophes need not be entered before and after the complete split-value.

- Split-value example: KEYWORD-value1/value2/value3
- Split-value sublist example: KEYWORD-'value1/value2, value1/value2'

Special Characters: Slashes (/), blanks, commas (,), hyphens (-), and apostrophes (') must not appear within the bounding apostrophes if a keyword is defined as either sublist or split-value. Any of these characters can appear within the apostrophes if the keyword is not defined as either sublist or split-value. In this case, a single apostrophe (') must be represented by two successive apostrophes ('').

Statement Length: Positions 1 through 71 of a record can be used. The first blank encountered following a keyword parameter, without indication of continuation to the next record, delimits the statement.

Continuation: Control statements can be continued on sub sequent input records by placing a comma immediately after a keyword parameter value on a statement. The comma must be followed by one or more blanks prior to or including column 72. (Thus the comma can appear in column 71 and a blank in column 72.) A continuation line must start with two slashes (//) followed by one or more blanks, followed by the remaining values (if part of a sublist) or the remaining keyword parameters. Any number of continuation records can be used. See Figure 6 for an example of coding.

Comments: The user can include, in addition to control statements, special statements that contain only comments. A comment statement is identified by an asterisk (*) in column 1 and can have any combination of characters. Any comments too long to be contained in 88 columns are truncated from the right.

In addition, comments can be included on any record of a control statement following the last parameter on that record. One or more blanks must separate the last parameter from the comments.

Special Meaning of Capital Letters, Numbers, and Special Characters: Capitalized words and letters, numbers, and special characters have special meanings in control statement descriptions:

- In control statements, capitalized words and letters must be written as they appear in the statement description. If numbers appear with the capitalized information, they must also be written as shown.

- Lower case terms indicate that a value must be substituted for each term.

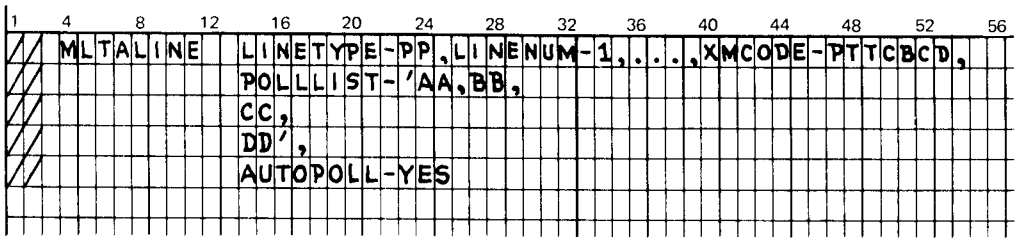


Figure 6. Continuation Control Statements

// SOURCE Statement

The SOURCE statement is optional and specifies that control statements supplied to $CCPAS are read from a source library member. When a source library is specified, one or more complete control statements must come from that source entry. After processing the last control statement in the specified source member, control statements are again read from the system input device. If the parameter END is specified with the value YES, $CCPAS goes to EJ. One or more SOURCE statements can be included anywhere within a control set but cannot be included within a source member.

```
// SOURCE  NAME-name, UNIT-{R1|F1|R2|F2}
           [,END-{NO|YES}  ]
```

NAME-name

This required parameter specifies the name of the source library member containing the control statements being supplied to $CCPAS. This member must have previously been built using $MAINT.

UNIT-{R1|F1|R2|F2}

This required parameter specifies the source library location.

END-{NO|YES}

This optional parameter indicates whether this source member contains the last control statement being supplied to $CCPAS. The default is NO.

// SET Statement

The SET statement identifies the assignment set and the operation to be done: create, replace, delete, or modify. This is the only statement required to delete an assignment set.

```
// SET     [ID-{1|c}] [,ACTION-{CREATE|REPLACE|DELETE|SYSMOD}]
           [,DFLTEXEC-{YES|NO}] [,ANYSPECS-{YES|NO}] [1]
```

ID-{1|c}

The ID parameter specifies which assignment set is to be affected. Each assignment set in $CCPFILE is identified by a different single character.

If the parameter is omitted, the default value is 1. Any single extended alphameric character (0-9, A-Z, #, $, @) can be used to identify the assignment set.

ACTION-{CREATE|REPLACE|DELETE|SYSMOD}

The ACTION parameter specifies what operation is to occur. If it is omitted, the default value is CREATE.

Note: If the set being deleted or replaced is the set CCP is executing from in another partition, the program use count for that execution of CCP will be lost (5704-SC2 only).

CREATE

If CREATE is used and the set ID does not exist in the file, the new set is added to the file. If the created set cannot fit, either because there is not enough room in the file for the tables or not enough room for the entry in the directory, an error message is given and the request is ignored. A maximum of 25 sets are allowed in the control file.

[1] Applies only to Program Number 5704-SC2.

REPLACE

If REPLACE is used and the set exists in the file, the set in the file is entirely replaced with the information on the statements following the SET statement. If the value is REPLACE and the set cannot be found in $CCPFILE, a warning message is given and the action defaults to CREATE.

DELETE

If DELETE is used and the set ID is in the $CCPFILE, the set and its table values are deleted from the file. If the value is DELETE and the specified set is not in the file, a warning diagnostic is given.

SYSMOD

The SYSMOD action indicates that only the SYSTEM statement, the execution default set identification, the any specifications identification, or all are to be modified in a set. The set must already be in the file (for further definition, see index entry *SYSTEM Statement*).

DFLTEXEC- { YES / NO }

The DFLTEXEC parameter specifies whether this control set ID is used as a default if no set ID is given at the start of execution of the CCP. If the parameter is not given, the default value is NO. If the value YES is given, then this set is used as the default at startup. The last processed SET statement that has this parameter and the value YES specified overrides any previous values.

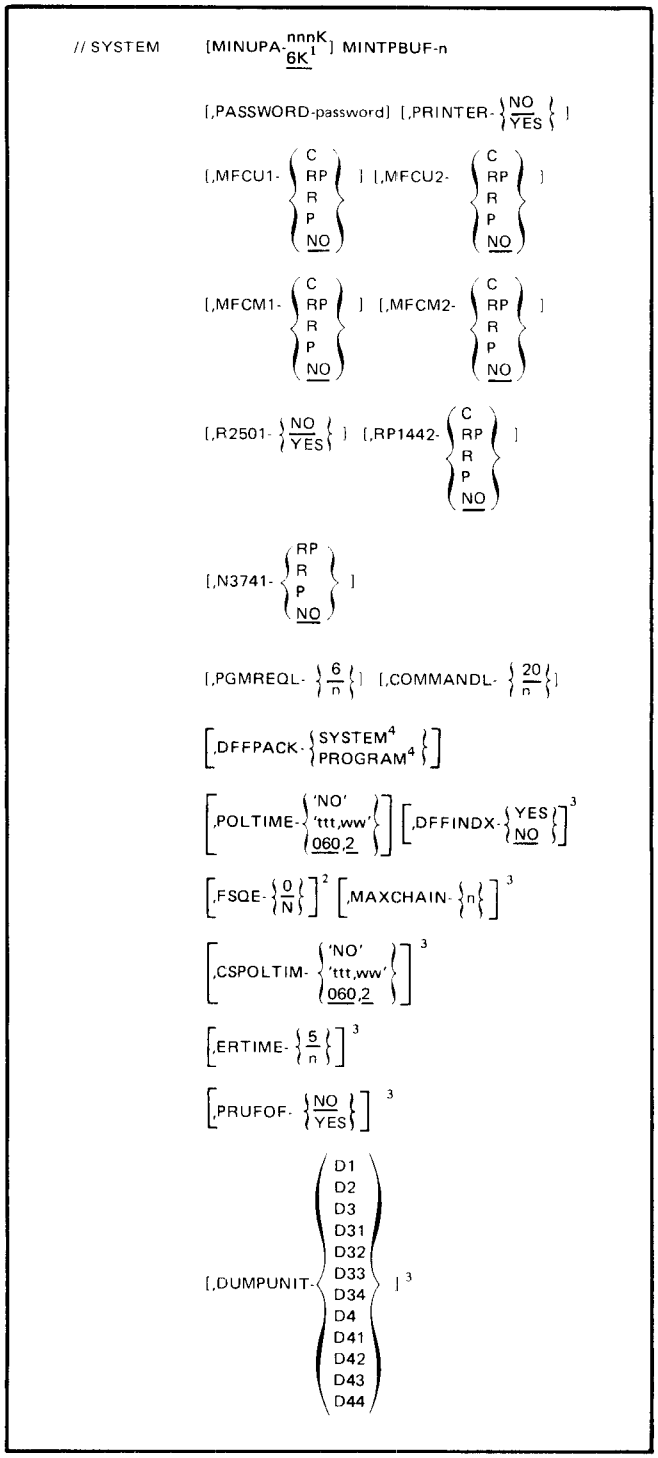
ANYSPECS- { YES / NO } (Program Number 5704-SC2 only)

The ANYSPECS parameter indicates whether the 'SU 01 ANY SPECIFICATIONS?' startup message is to be bypassed during CCP startup.[1] This parameter gives the ability to eliminate a startup prompt. If the $CCPFILE is located on the program pack, and a $CCPDUMP file is located on the unit specified during generation or on the unit specified by the DUMPUNIT parameter on the SYSTEM statement, a nonstop startup procedure is permitted. DFLTEXEC-YES must be specified on the SET statement to allow the ANYSPECS-NO parameter on the SET statement. The last processed SET statement that has these parameters with the value YES specified overrides any previous values.

// SYSTEM Statement

The system statement defines several facts about the system environment in which this assignment set is used:

- Minimum total main storage space required for the user programs which must execute under this assignment set.
- Whether the printer, MFCU, 2560, 2501, 3741, and 1442 are initially allocated to the CCP program level.
- Password for this set, if the user chose the option at CCP generation.
- The number of bytes set aside as the CCP dynamic buffer area for communicating with terminals.
- Maximum length of commands and program requests from terminals other than the console.
- Pack on which the user's DFF resides.



[1]Default for 5704-SC2 is 8K.
[2]Does not apply to Program Number 5704-SC2.
[3]Applies only to Program Number 5704-SC2.
[4]For Program Number 5704-SC2, these codes may also be specified:
R1, F1, R2, F2
D1A, D1B, D1C, D1D
D2A, D2B, D2C, D2D
D3A, D3B, D3C, D3D
D3E, D3F, D3G, D3H
D4A, D4B, D4C, D4D
D4E, D4F, D4G, D4H

It is possible to modify only the SYSTEM statement for a set which already exists in the $CCPFILE. If ACTION-SYSMOD was specified on the set statement, the system statement must be the only other statement given for this set. In this case, only those parameters given on this statement are used to modify the parameters that were given when the set was created, replaced, or last modified.

MINUPA-nnnK

The MINUPA parameter specifies the minimum size of user main storage area for executing programs in an operational run of the CCP for this set. The size of user main storage, when using DFF, must be large enough to include the size of the work space added to a DFF program. This size is determined by the DFFSFDT, DFFMTERM, and DFFNDF parameters in the program statement. The sizes associated with the DFF can be determined from *Disk Storage Estimates for CCP* in Appendix F.

If less than nnnK is left at the end of startup, a message is issued with the cancel option.

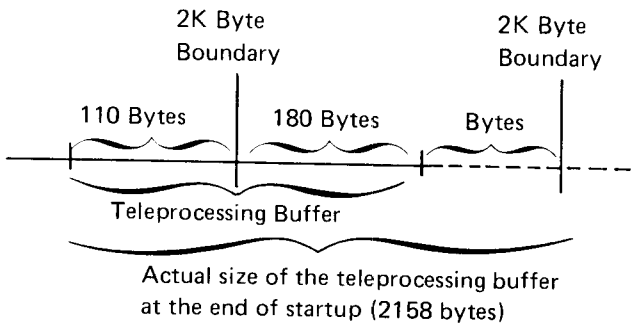
The value for this parameter must be one to three digits and must end with the letter K. K represents 1024 bytes. The value must be a multiple of 2K. The minimum (and default) values are 6K for Program Number 5704-SC1, and 8K for Program Number 5704-SC2. The maximum value is 256K (5704-SC1) and 512K (5704-SC2).

MINTPBUF-n

The MINTPBUF parameter specifies the number of bytes that must be set aside for buffer work area for teleprocessing communications. For an example of determining the size of the teleprocessing buffer, see index entry: *sample assignment set; calculation of main storage sizes.* The TP buffer area is calculated using the formula contained in Appendix F.

The value specified for this parameter (or changed during startup) is rounded up to the next multiple of 2K by start-up. The actual value depends on the size of the internal control tables built by startup. For example:

where MINTPBUF-290 is specified



PASSWORD,password

The PASSWORD parameter specifies the system password. This parameter must be specified if the SECURE-CCP option was selected on the $ESEC control statement at generation, and must not be specified if that option was not selected. If the user chooses to use his own security system, then this parameter must be omitted (the security information he checks a sign-on against is set by using the program $CCPAU).

If this parameter is given, the value must contain one to six characters specifying the system password. The password can be any combination of the 64-character set with no embedded blanks.

If the value given for this parameter is a special character, it must be enclosed in apostrophes. (Special characters are any characters other than the 36 alphameric characters.) In order to code an apostrophe as a character in the password, two successive apostrophes must be coded.

PRINTER- {NO / YES}

The PRINTER parameter declares whether the printer is allocated exclusively to the program partition in which CCP is executing at the beginning of a CCP run. If the parameter is not given, the default is NO. If YES is specified, the printer must not be allocated to the other partition after the beginning of the CCP run (unless print spooling is used for both partitions). If NO is specified and the program statement has declared it as a required device, CCP allocate attempts to obtain the device from DSM when the program is requested.

MFCU1 / MFCU2 / MFCM1 / MFCM2 - {C / RP / R / P / NO}

These optional keywords specify whether the MFCU and MFCM are permanently allocated to the partition where CCP is executing. In addition, these keywords specify how the MFCU and MFCM hoppers are allocated by CCP start-up. The following explanations apply:

Parameter	Explanation
C	Combined file: Specify C if any program running under CCP using this assignment set references this hopper as a combined file.
RP	Reader/punch: Specify RP if combined file does not apply and programs running under CCP using this set use this hopper for input and output.
R	Reader only: Specify R if combined file or reader/punch does not apply and programs running under CCP using this set use this hopper as input.
P	Punch only: Specify P if combined file or reader/punch does not apply and programs running under CCP using this set use this hopper as output.
NO	If NO is specified with MFCU1, MFCU2, MFCM1, MFCM2, or omitted, the MFCU and MFCM are not permanently allocated to the CCP partition. The device can still be used but contention might occur with the other partition.

Note: If the specified hopper is not being spooled under CCP, the device might not be available to the other partition for spooling. For spooling considerations, see *IBM System/3 Model 15 User's Guide to Spooling,* GC21-7632.

R2501- {NO / YES}

This parameter declares whether the 2501 is permanently allocated to the partition where CCP is executing. If the YES option is chosen, the 2501 is not available to the other partition. If NO is specified and the program statement has declared it as a required device, CCP allocate attempts to obtain the device from DSM when the program is requested.

RP1442- { C / RP / R / P / NO }

This optional keyword specifies whether the 1442 is to be permanently allocated to the partition in which CCP is executing. If this keyword is specified as NO or omitted, the 1442 is not permanently allocated to the CCP partition. In addition, this keyword also specifies how the 1442 is allocated by CCP startup. The following explanations apply:

Parameter	Explanation
C	Combined file: Specify C if any program running under CCP using this assignment set reference this hopper as a combined file.
RP	Reader/punch: Specify RP if combined file does not apply and programs running under CCP using this set use this hopper for input and output.
R	Reader only: Specify R if combined file or reader/punch does not apply and programs running under CCP using this set use this hopper as input.
P	Punch only: Specify P if combine file or reader/punch does not apply and programs running under CCP using this set use this hopper as output.
NO	If NO is specified, the 1442 is not permanently allocated to the CCP partition.

Note: If the 1442 is not being spooled under CCP, the 1442 might not be available to the other partition for spooling. For spooling considerations, see *IBM System/3 Model 15 User's Guide to Spooling*, GC21-7632.

N3741- { RP / R / P / NO }

This optional keyword specifies whether the 3741 (as a unit record device) is to be permanently allocated to the partition in which CCP is executing. If this keyword is specified as NO or omitted, the 3741 is not permanently allocated to the CCP partition. In addition, this keyword also specifies how the 3741 is allocated by CCP startup. The following explanations apply:

Parameter	Explanation
RP	Input/Output: Specify RP if programs running under CCP using this set this device for input and output.
R	Input only: Specify R if input/output does not apply and programs running under CCP using this set are using a 3741 as an input device.
P	Output only: Specify P if input/output does not apply and programs running under CCP using this set are using a 3741 as an output device.
NO	If NO is specified, the 3741 is not permanently allocated to the CCP partition.

Note: If the 3741 is not being spooled under CCP, the 3741 might not be available to the other partition for spooling. For spooling considerations, see *IBM System/3 Model 15 User's Guide to Spooling,* GC21-7632.

PGMREQL- { 6 / n }

The PGMREQL parameter specifies the length, in bytes, of the longest program request command from any terminal used in this set that does not use PRUF. (See the PRUFLNG parameter on the // PROGRAM statement for PRUF request.) (CCP might allow more than the specified value.) This length includes program name, one intervening blank, and the data accompanying the program request if this option is used. The maximum value allowed is 80 bytes for Program Number 5704-SC1 and 255 bytes for Program Number 5704-SC2. The maximum of 255 bytes for Program Number 5704-SC2 can be used with CPU-CPU or MLTA applications, but the recommended maximum for 3270 and other environments is 80. If this parameter is not given, the default is 6 positions. The minimum value allowed is 2 positions.

Note: The BLKL parameter of the TERMATTR must be greater than or equal to PGMREQL.

COMMANDL- { 20 / n }

The COMMANDL parameter specifies the length, in bytes, of the longest terminal command, excluding the program request command, that can be entered from any command terminal used in this set. (CCP might allow more than the specified value.) The maximum value allowed is 80 characters. If this parameter is not given, the default is 20 characters. The minimum value allowed is zero characters.

DFFPACK- { SYSTEM[1] / PROGRAM[1] }

The DFFPACK parameter specifies the pack whose object library contains the display formats for this set. The display formats must be previously produced through the display format generator routine ($CCPDF).

If this parameter is not specified, the use of the display format facility (DFF) is not allowed for this set. All other parameters referring to the DFF are diagnosed as invalid.

DFFINDX- { YES / NO } (Program Number 5704-SC2 only)

The DFFINDX parameter specifies whether or not the DFF index resides in main storage. If YES is specified, an external pointer list (EPL) is built in main storage and the DFF indexes built in $CCPFILE are moved to the EPL.

POLTIME- { NO / 'ttt,ww' / 060 02 }

ttt = total time to poll (in seconds) continuously before initiating a wait.
ww = total time to wait (in seconds) before resuming polling (after interval polling has started).

POLTIME-NO indicates no interval polling for this assignment set. However, this can be overridden during execution (if the CCP was generated for interval polling) by the POLTIME command. In this case, INTPOL-YES must have been specified on the $EBSC generation statement.

The default for ttt is 060; the default for ww is 02.

CSPOLTIM- { NO / 'ttt,ww' / 060,02 } (Program Number 5704-SC2 only)

CSPOLTIM-NO indicates no interval polling for BSC lines 3 and 4 (BSCC) for this assignment set. However, this can be overridden during execution (if CCP was generated for interval polling) by the POLTIME command. In this case, INTPOL-YES must have been specified on the $ECSC generation statement.

ttt is total time (in seconds) that CCP should poll before initiating a wait. ww is the time (in seconds) that CCP should wait before resuming polling (after interval polling has started). The default for ttt is 060; for ww, it is 02.

[1] For Program Number 5704-SC2, these codes may also be specified:
R1, F1, R2, F2
D1A, D1B, D1C, D1D
D2A, D2B, D2C, D2D
D3A, D3B, D3C, D3D
D3E, D3F, D3G, D3H
D4A, D4B, D4C, D4D
D4E, D4F, D4G, D4H

FSQE- {0 / N}

The FSQE parameter (Program Number 5704-SC1 only) specifies the number (n) of file sharing queue elements to be built by CCP at startup time. If the default is used, CCP (at STARTUP time) calculates and builds the maximum number of elements that would ever be required under this assignment set, based on the following formula:

$$\text{number of shared files} \times \frac{\text{user program area}}{\text{program size}}$$

This is calculated for each program in the assignment set and the maximum product is used. The maximum is limited by the following formula:

maximum number of shared files on any PROGRAM statement x number of CCP user tasks (SCP generation option)

To prevent the possibility of running short of FSQEs, it is recommended that the default be used.

ERTIME- {5 / N} (Program Number 5704-SC2 only)

The ERTIME parameter (Program Number 5704-SC2 only) specifies the time interval in minutes between retries to terminals in error recovery if AUTOERP is specified for that terminal on the BSCATERM statement. Valid entries are between 1 and 99 minutes. The default is 5 minutes.

PRUFOF- {YES / NO} (Program Number 5704-SC2 only)

The PRUFOF operand specifies whether the terminal on which PRUF data was entered should remain in PRUF mode until the next output operation or should be changed to a non-PRUF terminal after an ACCEPT input operation. YES indicates the terminal will be changed to a non-PRUF terminal after an accept input operation. NO indicates the terminal will remain in PRUF mode until the next output operation is complete. The default is NO.

MAXCHAIN-n

The MAXCHAIN parameter (for Program Number 5704-SC2 only) specifies the number (n) of control blocks to be built for task chaining and can be a value from 1 to 15. The value to specify for this parameter is the number of concurrent task chain requests that may be queued. The value of n should not be higher than is essential because unneeded control blocks will be built and main storage space will be wasted. If this parameter is not entered, the task chain facility is not supported. In this case, if a task chain operation is attempted, the program issuing the operation is canceled with a 3F termination code.

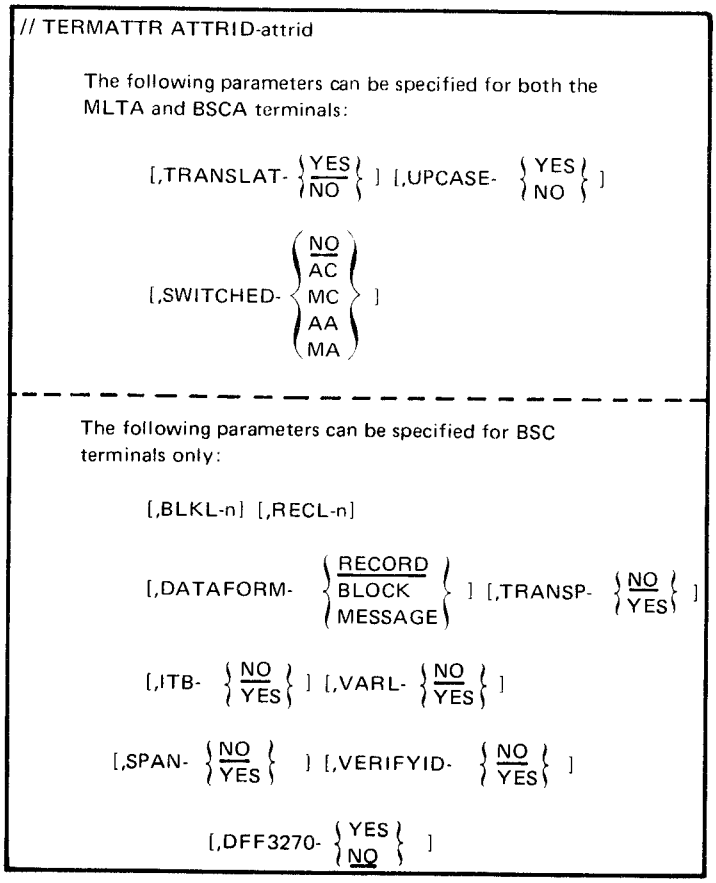
The value specified for this parameter plus the number of TERMNAME statements in this assignment set cannot exceed 254.

DUMPUNIT- {D1 / D2 / D3 / D31 / D32 / D33 / D34 / D4 / D41 / D42 / D43 / D44} (Program Number 5704-SC2 only)

This operand specifies the unit on which the CCP dump storage file ($CCPDUMP) resides. The DUMPUNIT parameter overrides the DPUNIT keyword specified on the $EGEN statement during CCP generation. If the DUMPUNIT parameter is not specified, CCP startup searches the unit specified during CCP generation.

// TERMATTR Statement

Each TERMATTR statement defines certain attributes of a terminal and must follow the system statement. The number of these statements required depends on the number and type of terminals used and how they are to perform in this assignment set. Each TERMATTR statement defines one terminal attributes set. One user program can refer to different attributes sets for the same terminal depending on the user of that terminal. Each terminal statement (BSCATERM and MLTATERM) must refer to a TERMATTR statement. One TERMATTR statement can provide the same attributes for several terminals and therefore be referred to by several terminals.



ATTRID-attrid

The ATTRID parameter must be specified. It is a reference ID number used in a BSCATERM or MLTATERM statement to associate this attributes set with a specific terminal. The value for this parameter can be any value between 1 to 255 inclusive. At startup time an amount of main storage (in bytes) equal to five times the highest number specified by any ATTRID parameter in this set is reserved for the entire terminal attributes table. Thus, the ATTRID values should be chosen from the smallest numbers available.

For 5704-SC2 only, ATTRID-04 must have specific attributes when programs such as CCPFMT, CCPPGM, or $QCOPY are to be used. Refer to the descriptions of these programs.

TRANSLAT- {YES / NO}

The TRANSLAT parameter specifies whether the CCP automatically translates input data from line code to EBCDIC, and output data from EBCDIC to line code. A YES value indicates translation takes place for MLTA transmission, and for BSC transmission using ASCII code.

A YES value is also valid if the line code is EBCDIC, though it causes no actual translation to occur unless UPCASE-YES is also specified. This would be a consideration when defining attributes for an EBCDIC 3270 terminal with a typewriter keyboard.

TRANSLAT-YES must be specified if using DFF with ASCII 3270 terminals.

A value of NO indicates the CCP does not do the translation; either the data coming in or going out is EBCDIC or the user handles the responsibility of translation to and from the appropriate line code.

UPCASE- {YES / NO}

The UPCASE parameter specifies whether lowercase characters transmitted from the terminal are converted to uppercase characters when presented to the user by the CCP. The YES value is valid only if the TRANSLAT parameter has the value YES. A value of YES indicates the CCP converts lowercase to uppercase before presenting data to the user program. A value of NO indicates the CCP does not convert lowercase characters to uppercase coming from the terminal. If not specified, UPCASE defaults to the same value as TRANSLAT. All input from terminals in command mode is translated to uppercase.

SWITCHED- {NO / AC / MC / AA / MA}

The SWITCHED parameter specifies the options available on switched lines. This parameter must be specified in an attributes set used with a terminal which is on a switched line. The value AC is not valid for MLTA lines. The default is NO. The following explanations apply:

Parameter	Explanation
NO	Not a switched line terminal.
AC	Automatic calling. The CCP calls the terminal using the telephone number provided on the TERMNAME statement.
MC	Manual calling. The system operator calls the terminal using the phone number printed by the CCP as provided on the TERMNAME statement.
AA	Automatic answering. The CCP answers calls from this terminal.
MA	Manual answering. The system operator answers the calls from this terminal.

MLTA, AA and MA are treated the same. For a command mode switched line, AC or MC must not be specified.

BLKL-n

The BLKL parameter defines the block length, in bytes, that is used for this terminal. It can be specified as 1 to 5 numeric digits with a valid range of 1 through 49152, but the actual value is limited by available main storage. This parameter is required when specifying attributes for any BSC terminal. The value used here must be equal to RECL times the number of records in a block. Up to 255 records can be contained in one block.

If BSCA terminal online tests are to be run using this attributes set, a value of at least 300 must be specified for BLKL. If less space is available, the test request is passed as data to the user program or the CCP command processor.

If OUTPOLL-YES is specified, the block length must be large enough to accommodate the outboard poll list. Two bytes per poll entry in POLLLIST are required. If there are 127 entries, then BLKL-254 would be the minimum block length.

The following are 3270 display formatting considerations:

- The value for this parameter can vary from 1 to 5120 according to the performance desired. For best performance this size should be large enough to hold the largest output display format that uses this terminal attributes set, the size of which is printed by the display format generation routine. Performance cannot be enhanced by specifying a larger value.

- If the value is less than the largest format, display formats are broken into output blocks and performance is decreased. With blocking, the minimum required size is 512.

RECL-n

The RECL parameter defines the record length, in bytes, that is used for this BSCA terminal. The value must be specified as 1 to 5 numeric digits with a valid range of 1 to the value given for the BLKL parameter. This parameter is valid only if the DATAFORM parameter on this statement has the value RECORD specified.

DATAFORM- { RECORD / BLOCK / MESSAGE }

The DATAFORM parameter defines the format in which the CCP presents terminal input data to the user program.

RECORD indicates the CCP presents a portion of an input block as a complete record to the user program.

BLOCK indicates a complete block (possibly consisting of multiple records) is presented as a unit of data.

MESSAGE indicates that all the data between the STX and the EOT is presented as one unit of data. MESSAGE is required if the parameter, DFF3270, on this statement has a value of YES.

The default value is RECORD.

TRANSP- { NO / YES }

The TRANSP parameter specifies whether the EBCDIC transparency feature is used in transmission to this terminal. The value YES can be used only for a BSCA-LINE statement with the XMCODE parameter value of EBCDIC (must have the transparency feature installed on the adapter). The value YES is valid only if XPRNCY-YES was specified in the $EBSC or $ECSC generation control statement. The default, if the parameter is omitted, is NO, indicating EBCDIC transparency is not to be used when transmitting to this terminal. On transmissions from a terminal, the line control characters received determine transparency.

If a terminal sends transparent data, the adapter must have the transparency feature and XPRNCY-YES must be specified on the $EBSC or $ECSC generation control statement.

ITB- { NO / YES }

The ITB parameter specifies whether intermediate text blocks are sent or received when using this terminal. The value YES is valid only for BSCA terminals. If ITB is specified as YES (ITB-YES must also be specified on the $EBSC generation control statement) then the parameters VARL and SPAN on this statement must have the value NO. The default value is NO.

VARL- { NO / YES }

The VARL parameter specifies whether variable length records with record separator characters are used in data transmission to or from this terminal. The value YES is valid only for BSCA terminals. The value YES is not valid if the ITB parameter on this statement has a value of YES. The default is NO.

SPAN- { NO / YES }

The SPAN parameter specifies whether input records can span input blocks. The value YES is valid only for BSCA terminal type 3735 or CPU. The value YES is not valid if the ITB parameter on this statement has a value of YES. The default is NO.

VERIFYID- {NO / YES}

The VERIFYID parameter specifies whether the CCP verifies the identification bytes sent from this terminal. YES indicates the CCP will verify. NO, the default value, indicates the CCP will not verify the ID of this terminal. The value YES is only valid for BSCA terminals on switched lines.

Note: If using ID verification, all online terminals on a BSCA switched line should have VERIFYID-YES. If one terminal online has VERIFYID-NO, anyone can connect via a telephone call as the non-verified terminal.

DFF3270- {YES / NO}

The DFF3270 parameter specifies whether the Display Format Facility (DFF) will be used with all terminals referencing this terminal attributes set. This parameter is valid only for 3270 terminals on a BSCA line. If this parameter is specified, the parameter DATAFORM-MESSAGE must be specified in this statement, and the SYSTEM statement in the assignment set must have the DFFPACK parameter specified. The default value is NO.

// BSCALINE Statement

The BSCALINE statement defines the type of BSC line to be used and the features of the line. A BSCALINE statement must be followed by at least one BSCATERM statement. The BSCATERM statement defines the terminals that are attached to that line. If more than one communication line or adapter is attached to the System/3, another line statement can be given and follows the last BSCATERM or MLTATERM statement for the previous line. The order in which line statements are entered determines the order of priority during the CCP execution. At least one line statement must be given for each assignment set, either BSCALINE or MLTALINE.

```
// BSCALINE TYPE
{[PP] [MP] [SW]} [,IDEXSEND-EXCHNGID] { LINENUM- {1 / 2} }
{ CS }
POLL LIST-'TERMID[,TERMID]...', LINENUM[1]-
{1 / 2 / 3 / 4} [,OUTPOLL- {YES / NO}]}
[,XMCODE- { EBCDIC / ASCII } ]
[,POLLLIST- 'termid [,termid] . . .']
[,NRETRY- {7 / n} ] [,IDEXSEND—exchngid]
[,DBLBUF- { NO / YES } ] [,WAIT- {180 / n} ]
[,DFFBUF- {YES / NO}][2]
```

[1] Line numbers 1 and 2 are valid for System Control Program Number 5704-SC1. Line numbers 1, 2, 3, and 4 are valid for System Control Program Number 5704-SC2.
[2] Program Number 5704-SC2 only.

TYPE- {PP / CS / MP / SW}

The TYPE parameter must be given and defines how this line is used.

One of the following values must be specified:

Parameter	Explanation
PP	Point-to-point. Dedicated communication line (leased or private) network on this line between processing units is supported by the CCP. One, and only one, BSCATERM statement must follow this statement.
CS	Control station. Multipoint communication line (leased or private) network on this line is supported by the CCP. This processor is the control station. At least one but not more than 45 BSCATERM statements must follow this statement. The POLLLIST parameter on this statement must also be used. For BSCC support (LINENUM-3 and LINENUM-4) TYPE-CS or TYPE-PP is required.
SW	Switched. This is a telephone-connected communication line. One or more terminals can be specified but only one terminal can communicate with the CCP at a time. One or more BSCATERM statements must follow this statement.
MP	Multipoint tributary station. Multipoint communication line (leased or private) network on this line is supported by the CCP. This System/3 is a tributary to a controlling processing unit. One, and only one, BSCATERM statement must follow this statement and must describe this System/3 as a terminal.

LINENUM- $\left\{\begin{matrix}1\\2\\3\\4\end{matrix}\right\}$

The LINENUM parameter identifies which BSC line this line statement defines. The numbers 1 and 2 identify BSCA adapter numbers 1 and 2, respectively. The numbers 3 and 4 identify BSCC lines 1 and 2 respectively (Program Number 5704-SC2 only). If not specified, LINENUM defaults to line 1. Two BSCALINE statements, within one set, specifying the same line number are not accepted.

If this BSCALINE statement is defining the display adapter, the LINENUM parameter must specify line 2. BSCC assignment support (5704-SC2 only) is implemented when LINENUM-3 or LINENUM-4 is specified. LINENUM-4 requires that BSCC-2 be specified on the $ECSC generation macro statement (support for both lines).

XMCODE- $\left\{\begin{matrix}\underline{\text{EBCDIC}}\\\text{ASCII}\end{matrix}\right\}$

The XMCODE parameter defines the transmission code used by the terminals on this line. If not specified, XMCODE defaults to EBCDIC. EBCDIC is BSCA EBCDIC code. ASCII is ASCII code.

POLLLIST-'termid [,termid] . . .'

The POLLLIST parameter defines the order the terminals attached to this line are to be polled. This parameter is valid only for TYPE-CS lines. The values can be given as a sublist: IDs are separated by a single comma and the entire parameter value is enclosed by apostrophes. Any value within the list must be a termid of a terminal specified on a BSCATERM statement. The termid given in a POLLLIST parameter in this BSCALINE statement must not appear in the POLLLIST parameter of another BSCALINE statement. A terminal ID can appear more than once in the same POLLLIST parameter. The maximum number of all terminal IDs that can appear in a poll list is 127.

The poll list must be given for all lines with TYPE-CS.

Note: For printers attached to a 3271 control unit, the *termid* of the terminals must appear in the values for POLLLIST even though they are not input devices. The printers are polled for status only when the CCP detects an error condition.

NRETRY- $\left\{\begin{matrix}\underline{7}\\n\end{matrix}\right\}$

The NRETRY parameter defines the number of retries that take place if there is a transmission error on the line before an error condition is considered to be uncorrectable. A 1- to 3-digit value up to 255 is acceptable. If the parameter is omitted, the default value of 7 is assumed. If the error persists after the specified number of retries, the error is logged on the system console. If a program is in control of the terminal, then an error return code is returned to the user program. If the terminal is not under the control of a user program, the CCP permits the system operator to bypass or retry the operation.

IDEXSEND-exchngid

The IDEXSEND parameter specifies identification characters this CPU (running CCP) sends to a remote terminal (processing units, 3741, or 3735 terminals). This allows the remote terminal to verify that this processing unit is the one the remote terminal wants to communicate with. The identification characters can be specified as any 1 to 15 extended alphameric characters or any 2 to 30 hexadecimal characters. The identification characters must be expressed in the transmission code (EBCDIC or ASCII). If the code is ASCII, only the hexadecimal representation can be used. Two hexadecimal characters represent one byte, and therefore there must be an even number of hexadecimal characters specified. The value is identified as being hexadecimal by enclosing the value in asterisks.

Example: IDEXSEND-*C1C2C3C4C5*

The IDEXSEND parameter should be specified only if the remote terminal expects to receive ID exchange characters. This parameter must be used if the terminal(s) on this switched line is a 3735 terminal. Also, this parameter is valid only if the TYPE parameter is SW.

DBLBUF- {NO / YES}

The DBLBUF parameter specifies whether terminals on this line are double line buffered. YES indicates double buffering is provided by the CCP; the main storage is reserved at startup. NO indicates a single buffer is provided by the CCP. The default is NO. Double buffering generally improves data transmission time, especially in a multiple block transmission environment. This specification refers to the line buffer, not the buffer work area specified by the MINTPBUF parameter on the system assignment statement.

WAIT- {180 / n}

The WAIT parameter specifies a decimal delay count. The delay count is the number of seconds after receiving or transmitting a block of data that the CCP waits for the user to receive or transmit another block of data for the same file. The CCP waits the specified number of seconds by using the WACK ENQ and TTD NAK line control sequences A 1- to 4-digit value up to 9999 is accepted.

Except when the end of file is received or transmitted, the CCP terminates the transmission and posts a completion code if the delay count is exhausted between transmissions. If a value is not specified, a 180-second delay count is assumed. If a delay count is specified, consider the time that may be required for such items as device errors, halts, and ready I/O devices.

This parameter applies only when user programs are communicating with a terminal on this line.

DFFBUF- {YES / NO} (Program Number 5704-SC2 only)

The DFFBUF parameter specifies whether or not a separate 2K buffer is to be created for DFF user PUTs. YES indicates 2K of main storage will be reserved at the startup phase of CCP for this BSC line. NO indicates a 2K buffer will not be reserved and all PUTs will be done from the TP buffer. The default is NO.

(Buffering may potentially improve data transmission time and response time by allowing multiple user PUTs to be in the process of transmission at any given time. No advantage is gained if only one BSC line is using DFF.)

If performance is affected by maximum utilization of the TP buffer, specifying DFFBUF-YES may result in some performance improvement.

OUTPOLL- {YES / NO} (Program Number 5704-SC2 only)

The OUTPOLL parameter specifies whether or not outboard polling is supported for BSCC. This parameter is valid only with LINENUM-3 or LINENUM-4 and only if TYPE-CS is specified. The outboard polling feature is automatically incorporated at execution time if the active terminals are 3270 only. Mixed terminal environments require that OUTPOLL-YES be specified in order to incorporate the feature at all times.

// BSCATERM Statement

The BSCATERM statement defines certain attributes of the terminals on BSC lines. It also references terminal attributes sets that complete the terminal specifications. The BSCATERM statements must follow the BSCALINE statement for which the terminals are defined.

The number of BSCATERM statements allowed depends on the line type; this is defined by the TYPE parameter on the BSCALINE statement. The maximum number of terminals allowed in a set is 120.

In addition, the maximum number of terminals per line type is:

- Control station line—45 terminals
- Direct attached line—30 terminals

```
// BSCATERM  TERMID-termid,TYPE-termtype,
               ATTRID-'attrid[,attrid] . . .'

          ,COMMAND- {NO}  [,ONLINE- {YES} ]
                    {YES}           {NO}

          [,AUTOERP- {NO}  ] [1]
                     {YES}

          [,IDEXRCV-exchngid] [,OFFACTN- {HOLD} ]
                                         {DROP}

          [,ADDRCHAR—addressing characters]

          [,POLLCHAR—polling characters]

          [,ICF- NO  ]
                 YES
```

[1] Program Number 5704-SC2 only.

TERMID-termid

The TERMID parameter assigns permanent (within this set) identification characters to the terminal. The valid entries for this keyword are two nonblank extended alphameric characters and are the terminal ID referred to by the system operator. There is no default for this parameter. The value given must be unique for each terminal within this set. The value must not be the reserved ID, $C.

TYPE-termtype

The TYPE parameter specifies the terminal type the BSCATERM statement describes. For all lines, except BSCA switched lines, all terminals on a line must be of the same type. (All components of a 3270 system are considered to be of the same type.) Switched BSCA lines may have CPU, 3741, 3275 switched, and 3735 terminals on the same line. For Program Number 5704-SC2 only, terminal types on BSCA/BSCC control station lines may have 3270, CPU, and 3741 terminals on the same line. CPU indicates all processing units (S/360, S/370, S/7, S/3) capable of receiving or transmitting over binary synchronous communication lines, with the proper program support. There is no default for this parameter.

The following terminal types can be specified:

3275M1
3277M1
3284M1
3286M1
3275M2
3277M2
3284M2
3286M2
3735
3741
CPU

Note: A BSCATERM statement is not allowed for the 3284 Model 3 attached to a 3275. A 3735 type terminal on a nonswitched point-to-point line is supported as being on a control station line. The following terminals are supported in compatibility mode. Specify the appropriate entry for the terminal type as follows:

Terminal Type	Entry
3276	3277M2
3278	3277M2
3287	3284M2 or 3286M2
3288	3286M2
3289	3286M2
5231-2	3741

ATTRID-'attrid [,attrid] . . . '

The ATTRID parameter specifies which terminal attributes sets (defined by the TERMATTR statement) this terminal can use. The first attributes set is used in the following cases:

- At startup.
- When the terminal requests a CCP user program.
- When required by a user program and if the attrid specification is omitted in the TERMS parameter on the assignment program statement.
- If no other specification is given when a user program acquires this terminal during execution.

Attributes sets listed after the first are eligible for assignment to this terminal at the following times:

- When the terminal is required by a user program and attrid is specified in the TERMS parameter in the assignment program statement.
- When the user program acquires the terminal via the acquire and set terminal attributes operation.

The values given must correspond with a value given for the ATTRID parameter on a TERMATTR statement. In addition to the specifics given on this statement, the terminal attributes set supply certain other specifics for this terminal. There is no default for this parameter.

The ATTRID parameter can refer to a maximum of 32 attributes sets. All the sets referenced are considered in calculating the maximum line block size for BSCA lines.

COMMAND-{NO / YES}

The COMMAND parameter specifies whether this terminal is capable of requesting programs. If this parameter is given with the value YES, then the terminal must have both input and output capability. Any terminal can be specified as not being command capable; but, if the parameter is given with the value YES and the terminal does not have both input and output capability, an error diagnostic is given.

YES is invalid for 3735 or 3741 terminals. There is no default for this parameter.

If command terminals are on a switched line, the attributes set must specify either automatic answering or manual answering.

ONLINE-{YES / NO}

The ONLINE parameter is used to specify whether the terminal is available to be used at the beginning of a CCP run. The default value for the omitted parameter is YES, indicating it is available.

A terminal can be connected to the system, yet not be available because it is in the process of being serviced, or because no authorized person is presently at the terminal to use it. A value of NO indicates that this terminal should be treated by the CCP as if it were logically offline at startup.

This specification can be overridden during the execution of the CCP by the system operator.

AUTOERP-{NO / YES} (Program Number 5704-SC2 only)

The AUTOERP parameter selects the option of automatic retries for terminals in error recovery. The retries to the terminal will occur at the time interval selected via the ERTIME parameter on the SYSTEM statement. AUTOERP requires full timer support to be selected during SCP generation. YES specifies that automatic retries will be done; NO specifies that they will not be done. The default is NO.

IDEXRCV-exchngid

The IDEXRCV parameter specifies the verification characters the System/3 containing the CCP expects to receive when communicating with the proper remote switched station. The value for this parameter can be 1 to 15 extended alphameric characters or 2 to 30 hexadecimal characters. The identification characters must be expressed in the transmission code (EBCDIC or ASCII). If the code is ASCII, only the hexadecimal representation can be used. Two characters represent one byte, therefore there must be an even number of hexadecimal characters specified. The value is identified as being hexadecimal by enclosing the value in asterisks.

Example: IDEXRCV-*C1C2C3C4C5*

This parameter should only be specified if the remote terminal will send ID exchange characters.

OFFACTN- {HOLD / DROP}

The OFFACTN parameter applies only to command terminals (COMMAND-YES) and specifies the action to be taken with this terminal when the terminal operator issues the sign-off command. The only two acceptable values are HOLD and DROP.

HOLD means the line continues to be monitored by the system after the sign-off command is given. HOLD is the default value for a nonswitched line.

DROP means the terminal is set in offline status after the sign-off command is given. The terminal is no longer monitored for input and, if the terminal is connected by a switched line, the line is disconnected. DROP is the default value for a switched line.

The terminal operator can override the parameter specified when using the sign-off command.

ADDRCHAR-addressing characters

The ADDRCHAR parameter is the hexadecimal character representation of the address characters in the line code used with this terminal The value for this parameter must be 4 to 14 hexadecimal characters. The identification characters must be expressed in the transmission code (EBCDIC or ASCII). For a multipoint line (that is, this processing unit is a tributary station), this parameter refers to the characters by which this processing unit is addressed from the host processing unit. This parameter is not valid for terminals on a switched line or a point-to-point line. The value is identified as being hexadecimal by enclosing the value in asterisks. For a description of the valid characters for System/3, see *IBM System/3 Multiline/Multipoint Binary Synchronous Communications Reference Manual,* GC21-7573. The terminal address and transmission code are physically wired at each terminal. Consult your IBM customer engineer for the exact code wired.[1]

POLLCHAR-polling characters

The POLLCHAR parameter is the hexadecimal character representation of the address characters in the line code to be used with this terminal. The value of this parameter must be 4 to 14 hexadecimal characters. The identification characters must be expressed in the transmission code (EBCDIC or ASCII). For multipoint line (that is, this processing unit is a tributary station) this parameter refers to the characters by which this processing unit is polled from the host processing unit. This parameter is not valid for terminals on a switched line or point-to-point line. The value is identified as being hexadecimal by enclosing the value in asterisks. For a description of the valid character for System/3, see *IBM System/3 Multiline/Multipoint Binary Synchronous Communications Reference Manual,* GC21-7573. General poll is not supported for 3270 type terminals.

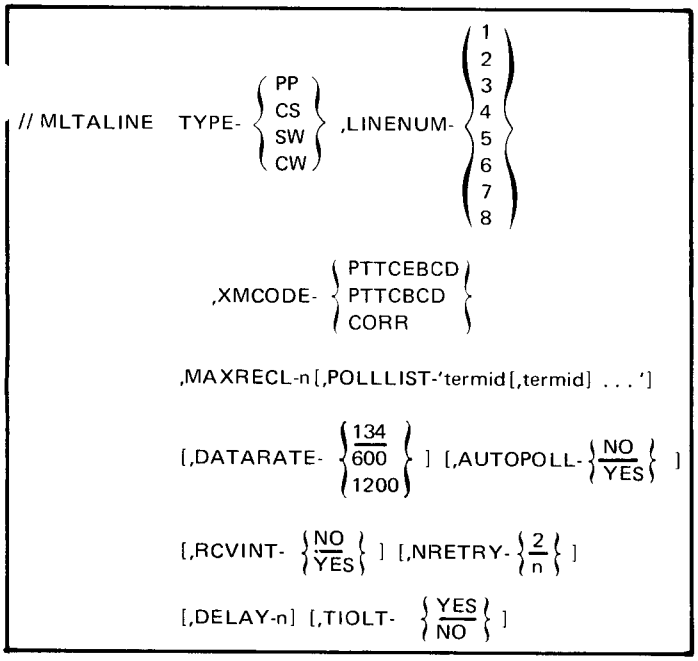
ICF- {NO / YES} (Program Number 5704-SC2 only)

The ICF parameter is used when a terminal is a System/34 using the ICF facility. Therefore, specify ICF-YES if the device is a System/34 with the ICF facility. The default is NO. Refer to the *System/34 Interactive Communication Feature Reference Manual,* SC21-7751.

// MLTALINE Statement

The MLTALINE statement is used to define the type of communication line to be used and the features of the line. A MLTALINE statement must be followed by at least one MLTATERM statement. The MLTATERM statement defines the terminals that are attached to that line. If more than one communication line or adapter is attached to the System/3, another line statement can be given and follows the last BSCATERM or MLTATERM statement for the previous line. The order in which line statements are entered determines the order of priority during the CCP execution. At least one line statement must be given for each set, either MLTALINE or BSCALINE.

[1] With BSC, no two terminals on the same line can have identical polling and addressing characters. With the display adapter (DA), no two terminals on the same line can have identical device characters. If using the display adapter (DA-YES in the $EBSC generation macro), the control unit addressing characters must be 6060xxxx and polling characters must be 4040xxxx.



TYPE- PP CS SW CW

The TYPE parameter defines how this line is used. There is no default.

The following values can be specified:

Value	Explanation
PP	Point-to-point. This is a dedicated communication line (leased or private). One, and only one, MLTATERM statement must follow this statement.
CS	Control station. This is a multipoint communication line (leased or private). This processor is the control station. All terminals on this line must have the station control hardware feature regardless of the number of terminals on the line. One or more MLTATERM statements must follow this statement.
SW	Switched. This is a telephone-connected communication line. One, and only one, MLTATERM statement must follow this statement.
CW	Switched with control station feature. This is a telephone-connected communication line (valid only for lines supporting 1050D terminals). One, and only one, MLTATERM statement must follow this statement.

LINENUM- 1 2 3 4 5 6 7 8

The LINENUM parameter identifies which MLTA line this statement defines. The numbers 1 to 8 identify the MLTA line numbers 1 to 8 respectively. There is no default for this parameter. Multiple MLTALINE statements, within one set, specifying the same line number are invalid.

XMCODE- PTTCEBCD PTTCBCD CORR

The XMCODE parameter defines the transmission code used on this line. The value given must correspond to the installed hardware feature on the terminal(s) to be used:

PTTCEBCD indicates paper tape transmission code EBCDIC
PTTCBCD indicates paper tape transmission code BCD
CORR indicates correspondence code

A 3767 used as a 2740M1 or 2740M2, and a 2741 uses PTTCEBCD or CORR.

A 5100 used as a 2741 uses PTTCEBCD or CORR.

MAXRECL-n

The MAXRECL parameter defines the maximum record length, in bytes, that is used on this line. It must be specified by giving one to three numeric digits. The specified length should not include room for line control characters. However, it does include any terminal device control characters that are used. For example, carriage return and idle characters inserted by the CCP in output to a typewriter terminal are included in the record (16 characters at the beginning and/or end of the message; the 2740 Model 2 with buffer receive requires one character at the beginning and/or end of the message). The valid range, for noncommand terminals, is 16 to 250 characters. The record length should not be less than the size of the largest record to be sent or received from an application program.

If any terminal on the line is a command terminal and that terminal is a 2740 Model 2, the minimum MAXRECL value is 77. If the command terminal is not a 2740 Model 2, the minimum MAXRECL value is 107.

Note: Online tests initiated by the system operator require a minimum record length of 100 bytes.

POLLLIST-'termid [,termid] . . . '

The POLLLIST parameter defines the polling order for the terminals attached to this line. The values can be given as a sublist; for example, terminal IDs are separated by a single comma and the entire sublist must be IDs of terminals specified on MLTATERM statements. The IDs given in the POLLLIST parameter must be unique for each line statement. (A particular ID cannot appear in the POLLLIST parameter of more than one MLTALINE statement in a set.) A terminal ID can appear more than once in the same POLLLIST parameter. The maximum total number of terminal IDs that can appear in a POLLLIST parameter is 127. For terminal types 1050 or 1050D, the maximum number is 31.

The poll list must be given for all lines when TYPE-CS is specified on this statement.

DATARATE- {134 / 600 / 1200}

The DATARATE parameter specifies the data rate of transmission this line is capable of handling; 134.5, 600, or 1200 bits per second (bps). A 1200 bps data rate is not available in the U.S.A. This parameter, if omitted, defaults to 134 (meaning 134.5 bits per second).

A 3767 used as a 2740M1 or a 2741 uses 300 bps. In this case, specify DATARATE 134. A 3767 used as a 2740M2 uses DATARATE 600 or 1200.

A 5100 used as a 2741 uses 300 bps. In this case, specify DATARATE 134.

AUTOPOLL- {NO / YES}

The AUTOPOLL parameter specifies whether the automatic polling feature is installed in the MLTA. Consult the IBM customer engineer to determine if this feature was installed. It can be installed if the 2740 Model 1 and Model 2 terminals have the station control feature, or if any terminals in the 1050 system are used. This parameter must be specified for every line statement if the feature is installed on the MLTA. If the parameter is omitted, the default is NO.

RCVINT- {NO / YES}

The RCVINT parameter specifies whether the receive interrupt feature is installed on the terminals. The default value is NO. The value YES is valid only for full duplex lines.

NRETRY- {2 / n}

The NRETRY parameter defines the number of retries that take place if there is a transmission error on the line. A 1- to 3-digit value with a maximum of 255 retries is acceptable. If the parameter is omitted, the default value is 2.

If the error persists after the specified number of retries, the error is logged on the system console. If a program is in control of the terminal, an error return code is returned to the user program. If the terminal is not under control of a user program, an error message is sent to the system operator.

DELAY-n

The DELAY parameter defines the delay time for control station type MLTA lines after the poll list is used and before polling starts again. This time is given in tenths of a second. If the number 15 were given, this would mean one and a half seconds.

The maximum value that can be given is 256 (25.6 seconds).

If this parameter is not specified, there is no delay.

TIOLT- {YES / NO}

The TIOLT parameter specifies whether the terminals on this line are allowed to request an online test. The default is YES. This parameter does not affect the capability of the system operator to initiate an online test.

NO should be specified only if five consecutive nines (99999) must be entered as data.

Notes:

1. No additional main storage is required to utilize the online test function.
2. Online test requires a minimum record length of 100 bytes.

// MLTATERM Statement

The MLTATERM statement defines certain attributes of the terminals on MLTA lines. It also refers to a terminal attributes set that completes the terminal specifications. The MLTATERM statements must follow the MLTALINE statement.

The number of MLTATERM statements allowed depends on the line type. This is defined by the TYPE parameter on the MLTALINE statement. The maximum number of terminals allowed in a set is 100.

```
// MLTATERM TERMID-termid,TYPE-termtype,
          ,ATTRID-'attrid [,attrid] . . . ' ,COMMAND- {NO / YES}
          [,ADDR-xx] [,ONLINE-{YES / NO}] [,PINCOMP-n]
          [,POUTCOMP-n] [,OFFACTN- {HOLD / DROP} ]
```

TERMID-termid

The TERMID parameter assigns permanent (for this set) identification to this terminal The valid entries for this parameter are any two (exactly two) nonblank extended alphameric characters. There is no default for this parameter. The value given must be unique for each terminal within this set. The value must not be the reserved ID, $C.

TYPE-termtype

The TYPE parameter specifies the terminal type this MLTATERM statement describes. All terminals on any MLTA line must be identical; (for example, 2740S, or 2740SC, or 1050). If a 1050 system is being described, refer to the parameters PINCOMP and POUTCOMP on this statement for further component description.

Any one of the following terminal types can be specified:

Character to Specify	Terminal and Features
1050	1050 on a nonswitched line.
1050D	1050 on a dial (switched) line.
2740	2740 Model 1 without additional features on a nonswitched line.
2740C	2740 Model 1 with longitudinal redundancy checking (LRC) feature on a nonswitched line (see note).
2740D	2740 Model 1 without additional extra features on a dial (switched) line.
2740DC	2740 Model 1 with longitudinal redundancy checking (LRC) feature on a dial (switched) line (see note).
2740DT	2740 Model 1 with transmit control feature on a dial (switched) line.
2740DTC	2740 Model 1 with transmit control and longitudinal redundancy checking (LRC) features on a dial (switched) line.
2740M2S	2740 Model 2 with station control feature on a nonswitched line.

Character to Specify	Terminal and Features
2740M2SB	2740 Model 2 with station control and buffer receive features on a non-switched line.
2740M2SC	2740 Model 2 with station control and longitudinal redundancy checking (LRC) features on a nonswitched line.
2740M2SCB	2740 Model 2 with station control, longitudinal redundancy checking (LRC), and buffer receive features on a nonswitched line (see note).
2740S	2740 Model 1 with station control feature on a nonswitched line.
2740SC	2740 Model 1 with station control and longitudinal redundancy checking (LRC) features on a nonswitched line (see note).
2741	2741 on a nonswitched line (see note).
2741D	2741 on a dial (switched) line (see note).
CMCSTD	Communicating Magnetic Card SELECTRIC® Typewriter on a dial (switched) line. The CMCST is supported to the extent that it functions identically to a 2741D.
SYS7C	System/7 functioning as a 2740 with longitudinal redundancy checking (LRC) feature on a nonswitched line. The System/7 is supported to the extent that it functions identically to a 2740C.
SYS7DC	System/7 functioning as a 2740 with longitudinal redundancy checking (LRC) feature on a dial (switched) line. The System/7 is supported to the extent that it functions identically to a 2740DC.
SYS7SC	System/7 functioning as a 2740 with station control and longitudinal redundancy checking (LRC) features on a nonswitched line. The System/7 is supported to the extent that it functions identically to a 2740SC.

Note: The 3767 terminal can be used under CCP to simulate the following MLTA terminals: 2740C, 2740DC, 2740M2SCB, 2740SC, 2741, 2741D. Restrictions on assignment when using the 3767 as one of these terminals are under the MCODE and DATARATE parameters of the MLTALINE statement.

The 5100 portable computer can be used under CCP to simulate the following MLTA terminals: 2741 and 2741D. Restrictions on assignment when using the 5100 as one of these terminals are listed under the XMCODE and DATARATE parameters of the MLTALINE statement.

ATTRID-'attrid [,attrid] . . . '

The ATTRID parameter specifies which terminal attributes sets (defined by the TERMATTR statement) this terminal can use. The first attributes set is used in the following cases:

- At startup.
- When required by a user program and if the attrid specification is omitted in the TERMS parameter on the assignment program statement.
- If no other specification is given when a user program acquires this terminal during execution.

Attributes sets listed after the first are eligible for assignment to this terminal at the following times:

- When the terminal is required by a user program and attrid is specified in the TERMS parameter in the assignment program statement.
- When the user program acquires the terminal via the acquire and set terminal attributes operation.

The values given must correspond with a value given for the ATTRID parameter on a TERMATTR statement. In addition to the specifics given on this statement, the terminal attributes set supplies certain other specifics for this terminal. There is no default for this parameter.

The ATTRID parameter can refer to a maximum of 32 attributes sets.

COMMAND- {NO / YES}

The COMMAND parameter specifies whether this terminal is capable of requesting programs. If this parameter is given with the value YES, then the terminal must have both input and output capability. Any terminal can be specified as not being command capable, but if this parameter is given with the value YES, and the terminal does not have both input and output capability, an error diagnostic is given. There is no default for this parameter.

ADDR-xx

The ADDR parameter is the two-hex character representation of the transmission line code of the terminal address. The terminal address and transmission code are physically wired at each terminal. Consult your IBM customer engineer for the exact code wired.

ONLINE- {YES / NO}

The ONLINE parameter is used to specify whether the terminal is available to be used at the beginning of a CCP run. The default value for the omitted parameter is YES. A value of NO indicates that this terminal should be treated by the CCP as if it were logically offline at startup. This specification can be overridden during the execution of the CCP by the system operator.

PINCOMP-n

The PINCOMP parameter is used only for 1050 terminals. This parameter specifies the input device for a 1050 system specified as a command terminal and indicates the component from which the CCP receives the terminal operator's input commands. For a noncommand terminal, either this parameter or the POUTCOMP parameter is required. Both can be specified if the terminal has both input and output capability.

The following numbers apply for the 1050 system:

1 indicates keyboard
2 indicates reader 1
3 indicates reader 2

POUTCOMP-n

The POUTCOMP parameter is used only for 1050 terminals, and specifies the principal (or only) output device of that terminal. This parameter is required for a 1050 system specified as a command terminal and indicates the component to which the CCP sends messages to the terminal operator. For a noncommand terminal, either this parameter, or the PINCOMP parameter is required. Both can be specified if the terminal has both input and output capabilities:

The following numbers apply for the 1050 system:

5 indicates printer 1
6 indicates printer 2
7 indicates punch 1
8 indicates punch 2

OFFACTN- {HOLD / DROP}

The OFFACTN parameter is used only for command terminals (COMMAND-YES). This parameter specifies the action to be taken with this terminal when the terminal operator issues the sign-off command. The only two acceptable values are HOLD and DROP.

HOLD means the line continues to be monitored by the system after the sign-off command is given. HOLD is the default value for a nonswitched line.

DROP means that, when the sign-off command is given:

- If the terminal is connected by a nonswitched line, that terminal's status is changed from online to offline.
- If the terminal is connected by a switched line, the line is disconnected.

DROP is the default for a switched line.

The terminal operator can override the parameter specified when using the sign-off command.

// PORTLINE Statement (Program Number 5704-SC2 only)

The PORTLINE statement defines how the BSCA, BSCC, or SIOC portline is to be divided into ports and the size of the line buffers. The PORTLINE statement should be entered after the SYSTEM statement or after the last TERMATTR statement. One PORTLINE statement may be entered for each BSCA, BSCC or SIOC line supported.

The PORTLINE statement follows this format:

```
// PORTLINE   MODE-{Primary  }
               {Secondary}

           TYPE-{SIOC    }
                {LINE1   }
                {LINE2   }
                {LINE3   }
                {LINE4   }
                {TTASK   }

           BLKL-{n  }
                {252}

           MAXMSG-n

           ,AQPORT-n

           ,NAQPORT-n

           [,PORTPRFX-portprefix]

           [,XMCODE-{EBCDIC}] [1]
                    {ASCII }

           [,NRETRY-{7}] [1]
                    {n}

           [,DBLBUF-{NO }] [1]
                    {YES}

           [,WAIT-{180}] [1]
                  {n  }
```

[1] BSC lines one through 4 only.

MODE-{{PRIMARY }}
     {{P   }}
     {{SECONDARY}}
     {{S   }}

This specifies which system is to be the PRIMARY and which system is to be the SECONDARY. It is the user's responsibility to ensure that one system is defined as PRIMARY and the other system is defined as SECONDARY. If both systems attempt to execute a PUT instruction simultaneously, the PRIMARY system will execute its PUT instruction first.

TYPE-{{SIOC }}
     {{S   }}
     {{LINE1 }}
     {{1   }}
     {{LINE2 }}
     {{2   }}
     {{LINE3 }}
     {{3   }}
     {{LINE4 }}
     {{4   }}
     {{TTASK }}
     {{T   }}

The TYPE parameter must be given to define what type of portline is being used.

One of the following values must be specified:

Parameter	Explanation
SIOC S	Serial input/output channel. Channel connected systems communication is supported by CCP. Allows CCP programs to pass information over an SIOC channel from one System/3 Model 15D to another System/3 Model 15D. See *IBM System/3 Model 15D Channel Connected Systems Program Reference and Logic Manual*, GC21-5199. (Program Number 5799-WNK is also required.)
LINE1 through LINE4 1 through 4	Portline over BSC lines. Allows CCP programs written to communicate over an SIOC channel to also communicate over a BSC line.
TTASK T	Task-to-task. Allows CCP programs to pass information from one CCP task to another CCP task within the same System/3 Model 15D. This is like SIOC tasks passing information over the SIOC channel to another SIOC task executing in another system.

BLKL-{n }
     {252}

The BLKL parameter defines the block length, in bytes, for the SIOC channel buffers. It can be specified as one to five numeric digits with a valid range of 252 to 49152 (all necessary IOBs will be added to the length to enable the block to transmit). The parameter defaults to 252 if not specified. This parameter is not used for PORTLINE operations other than SIOC.

MAXMSG-n

The MAXMSG parameter specifies the maximum number of bytes of the largest message to be received. It is specified as a number from 1 to 49152. This parameter must be specified. For PORTLINE other than SIOC, this value will be the block length.

AQPORT-n

The AQPORT parameter specifies the number of acquirable ports for this assignment set. The maximum number is 49; the default is 0.

NAQPORT-n

The NAQPORT parameter specifies the number of nonacquirable ports for this assignment set. The maximum number is 49; the default is 0.

At least one acquirable or one nonacquirable port must be specified. The number of acquirable ports on the primary system must match the number of nonacquirable ports on the secondary system. The number of nonacquirable ports on the primary system must also match the number of acquirable ports on the secondary system. When TYPE-TTASK is specified in this PORTLINE statement, the number of nonacquirable and acquirable ports must be equal.

PORTPRFX-portname

The optional PORTPRFX parameter defines the first characters of the generated portnames. The portprefix must be one to four alphanumeric characters.

The name of each logical port is generated using the characters given on the PORTPRFX parameter and appending a two digit port number. The acquirable port will start with the number 01 and the nonacquirable ports will start with the number 51.

XMCODE-{EBCDIC / ASCII}

The XMCODE parameter defines the transmission code used by the terminals on this line. If not specified, XMCODE defaults to EBCDIC. This parameter is valid only for TYPE-LINEn, where n=1, 2, 3, or 4.

NRETRY-{7 / n}

The NRETRY parameter defines the number of retries that take place if there is a transmission error on the line. If the error persists after the specified number of retries, the error is logged on the system console. If a program is in control of the terminal, an error return code is returned to the user program. If the terminal is not under the control of a user program, CCP permits the system operator to bypass or retry the operation. A one to three digit value up to 255 is acceptable. If the parameter is omitted, the default value of seven is assumed. This parameter is valid only for TYPE-LINEn, where n=1, 2, 3, or 4.

DBLBUF-{NO / N / YES / Y}

The DBLBUF parameter specifies whether terminals on this line are to be double line buffered. YES indicates double buffering is provided by CCP; the main storage is reserved at startup. NO indicates a single buffer is provided by CCP. The default is NO. Double buffering generally improves data transmission time, especially for multiple block transmissions. This specification refers to the line buffer, not the buffer work area specified by the MINTPBUF parameter on the system assignment statement. This parameter is valid only for TYPE-LINEn, where n=1, 2, 3, or 4.

WAIT-{180 / n}

The WAIT parameter specifies a decimal delay count. The delay count is the number of seconds, after receiving or transmitting a block of data, that CCP waits for the user to receive or transmit another block of data for the same file. CCP waits the specified number of seconds by using the WACK ENQ and TTD NAK line control sequences. A one- to four-digit value up through 9999 is accepted.

If the delay count is exhausted between transmissions, CCP terminates the transmission and posts a completion code except when the end of file is received or transmitted. If a value is not specified, a 180-second delay count is assumed. If a delay count is specified, consider the time that may be required for such items as device errors, halts, and ready I/O devices.

This parameter applies only when user programs are communicating with a terminal on this line. The parameter applies only for PORTLINE TYPE-LINEn, where n=1, 2, 3, or 4.

// TERMNAME Statement

The TERMNAME statement defines symbolic names associated with terminals and subterminals (a subterminal is a component of a 1050 system). This statement can also associate a telephone number to a specific terminal connected by a switched line. A maximum of 254 TERMNAME statements can be given for one assignment set.

At assignment time, every terminal must have at least one symbolic name associated with it. If more than one symbolic name is associated to a specific terminal, the first one given is called the primary name and those following are secondary names.

The first TERMNAME statement that has both NAME and TERMID parameters given for a specific terminal is the primary name for this terminal. Additional TERMNAME statements with the same TERMID and a different NAME parameter define names that can be given to the terminal via the /NAME terminal operator command.

```
// TERMNAME    NAME-termname[,TERMID-termid]
               [,MSTRNAME-termname] [,INCOMP-n]
               [,OUTCOMP-n] [,PHONENUM-number]
```

NAME-termname

The NAME parameter specifies a symbolic name to be associated with either a terminal or an input and/or output subterminal. If the parameter TERMID is specified on this statement, this NAME is associated with a specific terminal at the beginning of a CCP run. If the parameter MSTRNAME is specified, this NAME is the name of an input and/or output subterminal.

The value for this parameter must be one to six extended alphameric characters; it cannot be all blanks, CONSOL, or ALL. It must be different from any other name given on any other TERMNAME statement.

This symbolic name can be given and not associated with a terminal at assignment time. In this case it could, during a CCP run, be assigned to a terminal by the system operator. Also, if this symbolic name is assigned to a terminal, by using the TERMID parameter on this statement, it can be reassigned to another like terminal by the system operator during a CCP run.

TERMID-termid

The TERMID parameter associates the terminal which has this two-character ID with the symbolic name given in the NAME parameter. Specifying this parameter implies the NAME parameter value is a terminal name and therefore the parameters INCOMP, OUTCOMP, and MSTRNAME cannot be given on this statement. The termid value must be a termid given for a terminal on a BSCATERM or MLTATERM statement.

MSTRNAME-termname

The MSTRNAME parameter is used to associate the subterminal with a terminal symbolic name. This parameter should only be used when describing subterminals and cannot be specified if the TERMID parameter is specified. If this parameter is specified, either the INCOMP and/or OUTCOMP parameters must also be specified. The value given here must match a value given as a terminal symbolic name on a previous TERMNAME statement.

When describing a 1050 system, this parameter will associate the subterminal specified on this statement to a 1050 system symbolic area.

INCOMP-n

The INCOMP parameter specifies an input subterminal (input component of a multicomponent terminal) and implies the name given in the NAME parameter is the symbolic name associated with this subterminal. If this parameter is specified, then the MSTRNAME parameter must also be specified to associate this subterminal to a terminal symbolic name.

The value for this parameter must be a single digit number. For the 1050 system the following numbers apply:

1 indicates keyboard
2 indicates reader 1
3 indicates reader 2
4 indicates any one input component of the polled line

OUTCOMP-n

The OUTCOMP parameter specifies an output subterminal (output component of a multicomponent terminal) and implies the name given in the NAME parameter is the symbolic name associated with this subterminal. If this parameter is specified, then the MSTRNAME parameter must also be specified to associate this subterminal to a terminal symbolic name.

The value for this parameter must be a single-digit number. For the 1050 system the following numbers apply:

5 indicates printer 1
6 indicates printer 2
7 indicates punch 1
8 indicates punch 2
9 indicates any or all output components of the polled system

PHONENUM-number

The PHONENUM parameter defines the telephone number for this terminal. A telephone number is required for switched lines using automatic calling or manual calling. The value given must be a string of numeric digits representing the exact telephone number needed to place the call to this terminal. This parameter can be specified only if the TERMID parameter is given on this statement. A maximum of 25 digits can be given for any one telephone number.

// DISKFILE Statement

The DISKFILE statement is used to describe the disk files that are used during the execution of this set. The information given on this statement must correspond with the actual type of file as it is defined in the program.

A DISKFILE statement must be given for each disk file used by any program in this set, specifying the name by which programs refer to this file. A maximum of 50 DISKFILE statements are allowed within one assignment set for Program Number 5704-SC1, and 192 DISKFILE statements for Program Number 5704-SC2. However, the DSM might restrict the number of files that can be opened during CCP startup. When Program Number 5704-SC2 is used and there are more than 50 files specified in the assignment set, an external pointer list (EPL) is built in main storage. The size of this EPL can be either 2K or 4K bytes. It is put in the user program area at startup time.

```
// DISKFILE    NAME-filename
                ,ORG-{C}
                     {D},RECL-n[,KEYL-n]
                     {I}
                [,KEYPOS-n] [,MSTRINDX-{YES}]1
                                       {NO }
                [,MIXSIZE-n]1  [,EXTENTS-n]2
```

[1] Not applicable for Program Number 5704-SC2.
[2] Not applicable for Program Number 5704-SC1.

NAME-filename

The NAME parameter specifies the name by which a program refers to the file. This name matches the name on an OCL // FILENAME-cccccccc, . . . statement to be included at startup (unless that file is suppressed).

To define a file more than once:

Assignment Statements
// DISKFILE NAME-x,ORG-D, RECL n
// DISKFILE NAME-y,ORD-I,RECL-n,KEYL-n,KEYPOS-n

Startup OCL
// FILE NAME-x,UNIT-D1,PACK-D1D1D1
// FILE NAME-y,UNIT-D1,PACK-D1D1D1,LABEL-x

The files can be two read files, or one read file and one update file.

ORG- {C / D / I}

The ORG parameter specifies how the file is organized on disk. There is no default for this parameter.

The following values can be specified:

C indicates consecutive (sequential)
D indicates direct
I indicates indexed

To use this parameter most effectively, you should match the keyword value in this parameter with the access value for the FILES keyword in the PROGRAM assignment statement. That is, when processing a direct file or consecutive file consecutively, use ORG-C. When processing a direct file or consecutive file by direct access, specify ORG-D. When the same file is accessed by both direct and consecutive methods, determine the ORG value by specifying the access method used most often to open this file.

When running multivolume files, ORG-C (sequential) is invalid.

When running CCP/Disk Sort Programs, ORG-C must be specified for the work file, and the output file.

RECL-n

The RECL parameter specifies the record length of the file on disk. The record length on disk and the record length for that file in a program must be the same. The value given for this parameter can be from 1 to 4096.

For a CCP/Disk Sort program, the work file requires RECL-256.

KEYL-n

The KEYL parameter (required for indexed files) specifies the length of the key in an indexed file. The values can be from 1 to 29 but must correspond to the key length as defined in the file on disk.

KEYPOS-n

The KEYPOS parameter (required for indexed file) specifies the location of the key within the record. Specifically, the position given must be the position of the first character of the key in the record. This value is not a displacement value. This parameter must be given for all indexed files and can be from 1 through 9999, but must correspond with the value used when the file was created.

MSTRINDX- {YES / NO}

The MSTRINDX parameter (Program Number 5704-SC1 only) specifies whether this file should have a resident master index. The parameter can be omitted and the MIXSIZE parameter can be used to specify a master index and the main storage allocation for it. Master index is not supported for the 5445, 3340, or 3344 disk files. This parameter is valid only if the ORG parameter has a value of I.

The MSTRINDX-YES is specified and the MIXSIZE parameter is omitted, a master index of one entry per index cylinder is built at startup.

Note: The disk track index of a 5445, 3340, or 3344 index file is used by programs running under the CCP that are identical to programs running under disk system management.

The default is NO, unless MIXSIZE-n is specified with a nonzero value indicating no master index within the CCP. This master index is external to the user's program and is the only master index used for this file when this program is run under the CCP. Thus, if space for a master index was included in the user program, it is not used under the CCP.

The master index is a table containing entries for tracks in the index portion of an indexed data file. Each entry contains a track address and the lowest key field associated within that track. The most efficient size for the master index is equal to the key field length plus two multiplied by the number of tracks in the file index. A master index can significantly reduce the amount of time needed to process an indexed file on a 5444.

MIXSIZE-n

The MIXSIZE parameter (Program Number 5704-SC1 only) specifies the number of bytes to be allocated to the resident master index for this file. The user can specify up to 16,383 bytes for the master index. However, the largest resident master index that can be used is considerably less than this. The minimum value that can be specified is the number specified for KEYL plus five. This parameter is valid only if the ORG parameter has a value of I specified.

The following chart indicates results of various combinations of the MSTRINDX parameter and the MIXSIZE parameter:

	MIXSIZE Parameter	
	Nonzero	**Omitted**
MSTRINDX Parameter		
YES	Valid, the specified number of bytes is used for master index	Valid, one entry per cylinder of index
NO	Invalid	Valid, no master index
Omitted	Valid, the specified number of bytes are used for master index	Valid, no master index

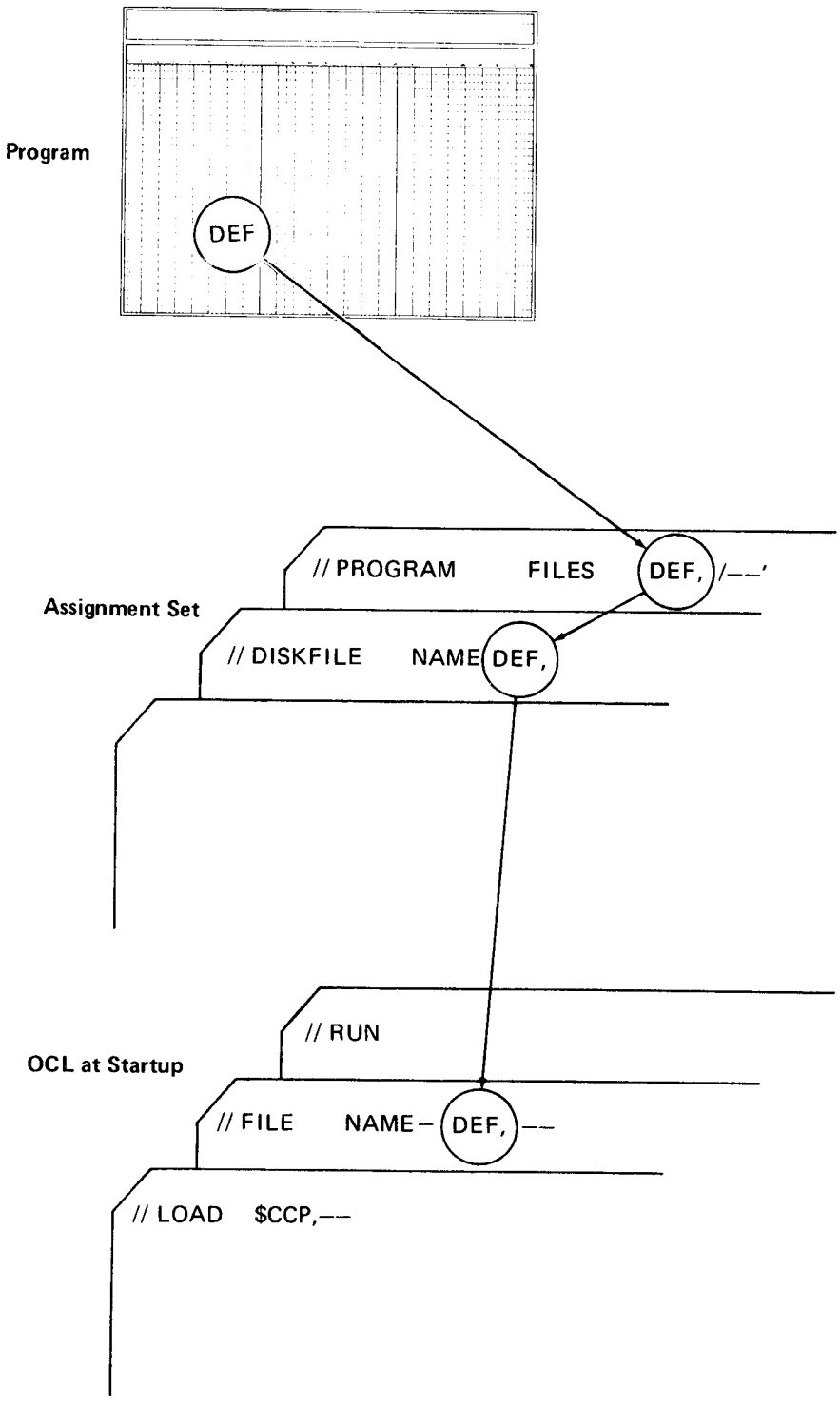
EXTENTS-n

The EXTENTS parameter (Program Number 5704-SC2 only) specifies the number of extents (volumes) for a multivolume file. The values can be 2, 3, or 4; the value must correspond to the number of extents indicated within the corresponding file OCL statement at startup.

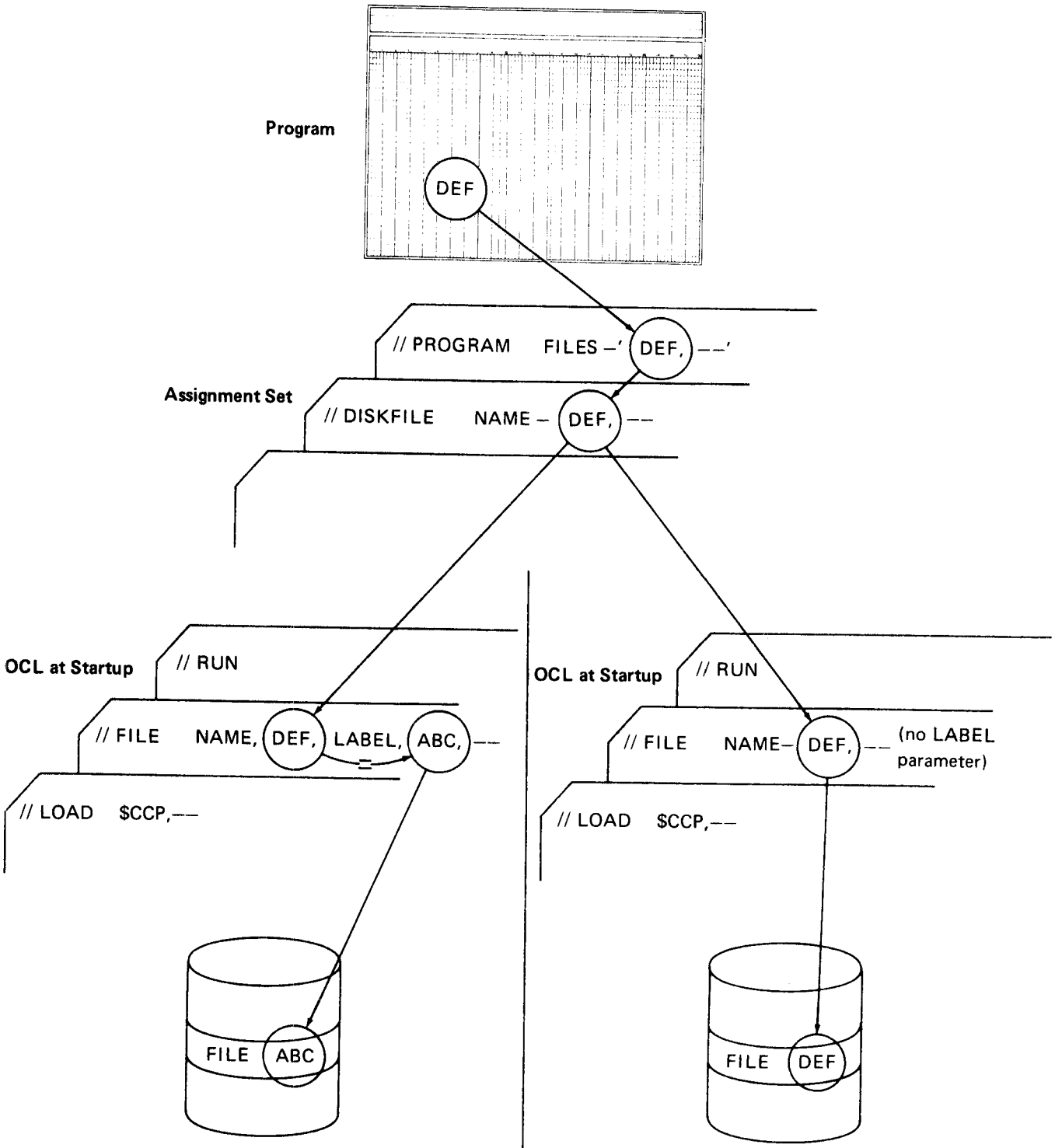
The filename used in the program . . . .

. . . . is the name that must be specified in a DISKFILE statement during assignment

. . . . and is the name that must appear as the NAME parameter in an OCL FILE statement at the startup of CCP.



Relationship of Filename Used in Program to Name of File on Disk



If the LABEL parameter is specified in the OCL FILE statement, the actual file referred to is the file that resides on disk under the name specified by that LABEL parameter.

If no LABEL parameter is specified in the OCL FILE statement, the actual file referred to is the file that resides on disk under the name specified by the NAME parameter.

// SYMFILE Statement

The SYMFILE statement defines the symbolic disk file reference name. This statement specifies the set of disk files that the symbolic name can be validly associated with by the terminal operator /FILE commands (see index entry: *file command*). A maximum of 50 SYMFILE statements are allowed within one control set.

```
// SYMFILE    NAME-ccccccc,DISKFILE-'ccc . . . [,ccc . . .] . . . '
```

NAME-cccccccc

The NAME parameter specifies a symbolic name that is used as a file reference name within one or more programs in this assignment set. This name does not identify a particular file (as a DISKFILE statement NAME parameter does). A file command from the terminal operator is required to specify which disk file is actually referred to in execution of the program. The file command must specify one of the disk files identified in this statement. It cannot duplicate any name on a DISKFILE statement or another SYMFILE statement.

DISKFILE-'ccc . . . [,ccc . . .] . . . '

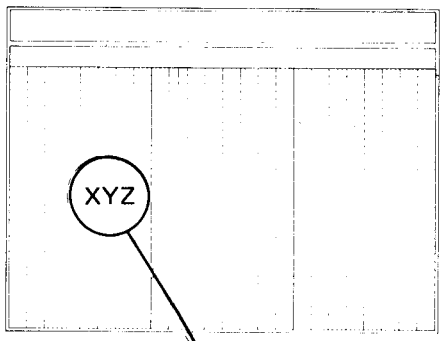
The DISKFILE parameter specifies one or more file names previously specified by the NAME parameter on a DISKFILE statement. All files referred to in the DISKFILE parameter of the SYMFILE statement must have identical characteristics. A symbolic file name cannot refer to both 5444 and 5445 disk files. For Program Number 5704-SC2 the maximum number of files is 191.[1]

SYMFILE statements should not be used to overlay input data with output data when running CCP/Disk Sort because unpredictable results may occur. Symbolic files cannot be validly defined for programs requested via the chain task process because there is no line of control between the requester of a chain task sequence and the programs in that sequence.

[1] Each SYMFILE statement in the assignment set takes up one file making the maximum number of files 191.

Symbolic Disk File Name

This filename used in a program is a symbolic filename.



Program

XYZ

The assignment set indicates that this name might refer to any one of three different disk files, depending on which one the terminal operator wants to use.

Assignment Set

// PROGRAM FILES–' XYZ /––'

// SYMFILE NAME – XYZ, DISKFILE – DEF, GHI JKL,

// DISKFILE NAME – DEF

// DISKFILE NAME – GHI

// DISKFILE NAME – JKL

// RUN

OCL at Startup

// FILE NAME – JKL,

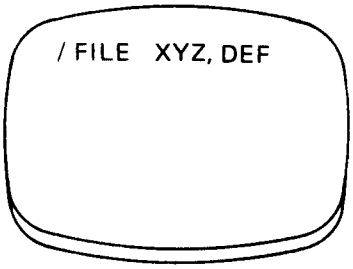
// FILE NAME – GHI,

// FILE NAME – DEF,

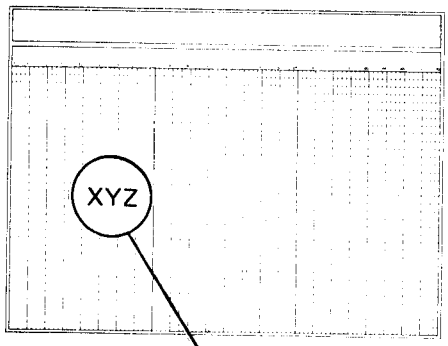
// LOAD $CCP,––

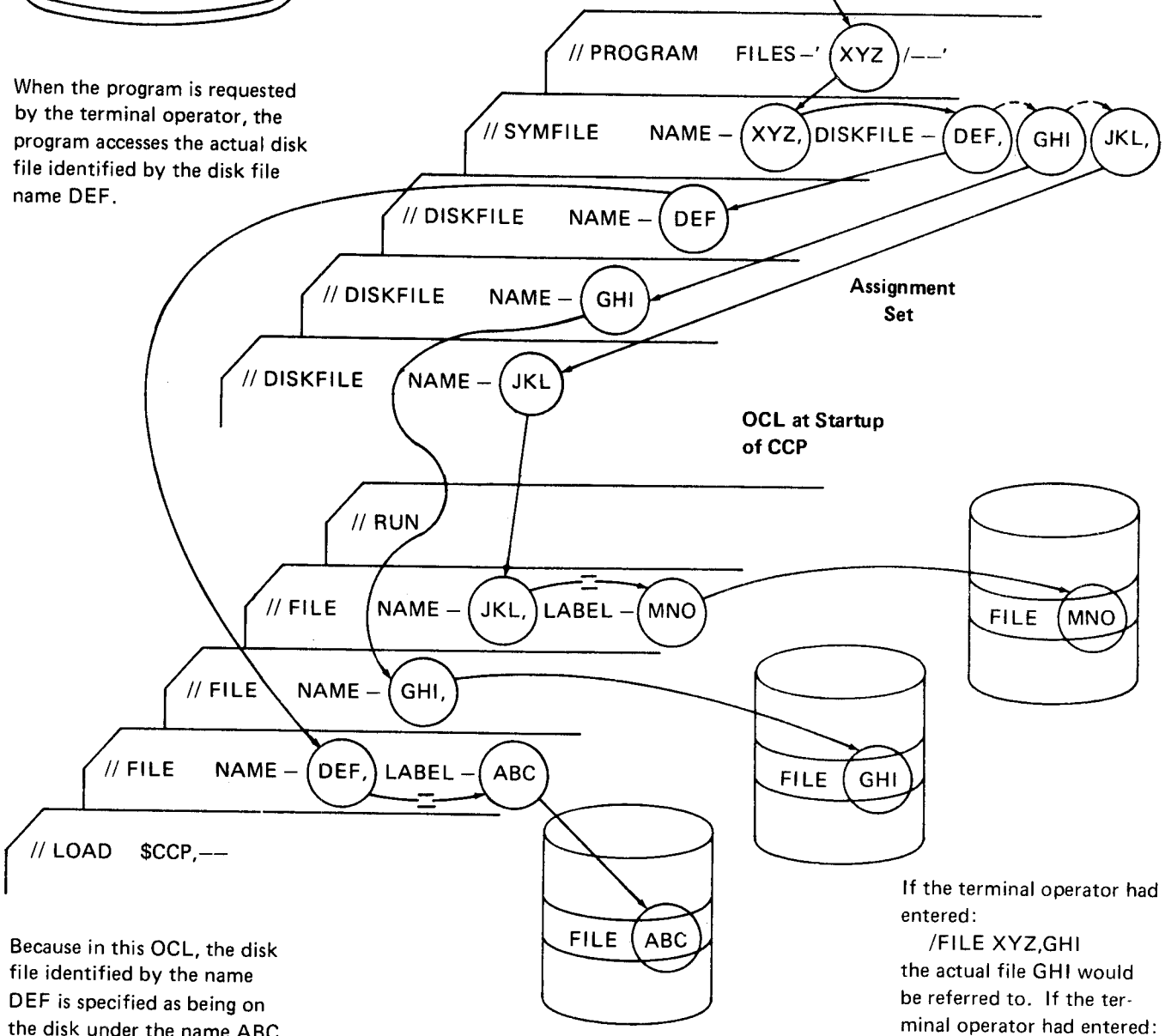
Each of the possible disk file names must appear as the NAME parameter of an OCL FILE statement at the startup of CCP.

Before running the program, the terminal operator entered this command:



When the program is requested by the terminal operator, the program accesses the actual disk file identified by the disk file name DEF.





Because in this OCL, the disk file identified by the name DEF is specified as being on the disk under the name ABC, the file ABC is the actual file referred to in this run of the program.

If the terminal operator had entered:

/FILE XYZ,GHI

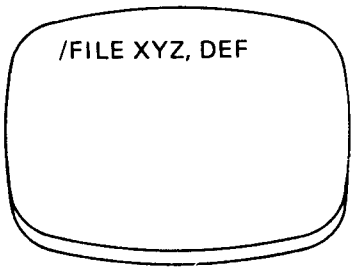
the actual file GHI would be referred to. If the terminal operator had entered:

/FILE XYZ,JKL

the actual file MNO would be referred to.

Two Filenames Used in a Program Refer to the Same File

Before running the program, the terminal operator entered this command:



By using a symbolic file, the same file is used under two names, perhaps in order to process the file in two different ways in the program.

// PROGRAM Statement

The PROGRAM statement defines the logical structure and the resource requirements of a user program. One PROGRAM statement must be given for each program that can be requested by a command terminal during the execution of the CCP using this assignment set. A maximum of 255 program statements[2] can be entered within one assignment set. This restriction does not apply to Program Number 5704-SC2; therefore the maximum number of programs is limited only by other application-dependent variables. For additional discussion about the programming languages supported, see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579.

Note: Unit record devices do not have to be specified on the // SYSTEM statement. It is dynamically allocated from DSM. However, if the device is not immediately available, the requested program is rejected.

```
// PROGRAM   NAME-pgmname
          [,MRTMAX-n] [,NEVEREND-{NO|YES}]
          [,PRINTER-{SHR|NO|YES}]
          [,MFCU1-{C|RP|R|P|NO}] [,MFCU2-{C|RP|R|P|NO}]
          [,MFCM1-{C|RP|R|P|NO}] [,MFCM2-{C|RP|R|P|NO}]
          [,RP1442-{C|RP|R|P|NO}] [,N3741-{RP|R|P|NO}]
          [,PGMDATA-{NO|YES}]
          [,ENDMSG-{NO|YES}] [,R2501-{NO|YES}]
          [,TERMS-'termname[/attrid] [,termname[/attrid]] ...']
          [,FILES-'filename/access[/{SHR|NOSHR|KSORT}],...']
          [,PACK-{PROGRAM[3]|SYSTEM[3]}]
          [,PRUF$Z-format name]
          [,PRUFLNG-nnnn]
          [,DFFMTERM-n] [,DFFNDF-n] [,DFFSFDT-n]
          [,MORCOR-n]
          [,SORT-{NO|YES}][1] [,EXECFIND-{NO|YES}][1]
          [,PRIORITY-{NORMAL|LOW}] [,TASKSIZE-n][1]
          [,PGMFIND-{NO|YES}][1]
```

[1] For Program Number 5704-SC2 only.
[2] For Program Number 5704-SC1 only.
[3] For Program Number 5704-SC2, these codes may also be specified:
R1, F1, R2, F2
D1A, D1B, D1C, D1D
D2A, D2B, D2C, D2D
D3A, D3B, D3C, D3D
D3E, D3F, D3G, D3H
D4A, D4B, D4C, D4D
D4E, D4F, D4G, D4H

NAME-pgmname

The NAME parameter defines the name of a program that resides on a user's CCP pack specified in the PACK parameter. The name must be one to six extended alphameric characters. However, it cannot start with a $ or a numeric character except as indicated in the following list:

$CCPDD
$CCPCL
$CCPOP

Exceptions:

- $CCPDD – can be included to dump all storage dumps from $CCPFILE or $CCPDUMP (5704-SC2) to printer while CCP is running. Include the following statement.

  // PROGRAM NAME-$CCPDD,PRINTER-YES,
  PGMDATA-NO

  For Program Number 5704-SC2 only, the $CCPDD program statement does not need to be included in the assignment set; a standard entry is made by the system program. If a program statement for $CCPDD is entered, it is ignored. This is to assure that $CCPDD will be available during execution of CCP.

- $CCPCL and $CCPOP – can be included to dynamically close and reopen disk files. Include the following statements:

  // PROGRAM NAME-$CCPCL,MRTMAX-1

  // PROGRAM NAME-$CCPOP

$FCOMP cannot move or back up data areas if the files have been opened or closed by $CCPCO, $CCPCL, or $CCPOP.

For a description of dynamic close/re-open, see *Disk File Close/Open Facility*.

- CCPFMT – can be included to find formats that have been added or updated since startup, and to update the DFFSFDT value (in the PROGRAM statement). For more information on CCPFMT refer to the *Format Find Routine* in the 3270 Display Format Facility (DFF) section of the *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579 or the *IBM System/3 Model 15 Communications Control Program System Operator's Guide*, GC21-7619.

MRTMAX-n

The MRTMAX parameter specifies that the program is capable of servicing multiple requesting terminals, and the number of terminals it is capable of servicing at one time. For a discussion about multiple requesting terminals, see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579. If this parameter is omitted, it indicates that this program (a single requester program) is not capable of servicing multiple requesting terminals. The valid range of numbers for this parameter is 1 to 239. MRTMAX-1 must be specified for the dynamic close facility, $CCPCL.

NEVEREND- {NO / YES}

The NEVEREND parameter specifies whether the program is never ending. The term *never ending* defines a program that, once it is initiated and loaded into main storage, remains in main storage for a relatively long time. If a resource conflict occurs and either the requester or owner is a never-ending program, the requester is rejected.

PRINTER- {NO / YES / SHR}

The PRINTER parameter specifies whether the line printer is used in the program. The default is NO. SHR indicates the program uses the printer but will share it with other programs running under CCP. However, if SHR is specified the output can be interleaved with other program's output.

R2501- {NO / YES}

This parameter specifies whether the 2501 is used in the program. The default is NO.

MFCU1
MFCU2
MFCM1
MFCM2 - { C / RP / R / P / NO }

These parameters specify how the MFCU and MFCM hoppers are used in this program. The following explanations apply:

Parameter	Explanation
C	Combined file. Specify C if this program uses this hopper for combined input and output.
RP	Reader/punch. Specify RP if combined does not apply and this program uses this hopper for both input and output.
R	Reader only. Specify R if this program uses this hopper for input only.
P	Punch only. Specify P if this program uses this hopper for output only.
NO	Default value if this hopper is not used in this program.

RP1442- { C / RP / R / P / NO }

This parameter specifies how the 1442 is used in this program. The following explanations apply:

Parameter	Explanation
C	Combined file. Specify C if this program uses this hopper for combined input and output.
RP	Reader/punch. Specify RP if combined does not apply and this program uses this hopper for both input and output.
R	Reader only. Specify R if this program uses this hopper for input only.
P	Punch only. Specify P if this program uses this hopper for output only.
NO	Default value if the 1442 is not used in this program.

N3741- { RP / R / P / NO }

This optional keyword specifies whether the 3741 (as a unit record device) is used by this program. The default is NO. The following explanations apply:

Parameter	Explanation
RP	Specify RP if this program uses the 3741 for both input and output.
R	Specify R if this program uses the 3741 as input.
P	Specify P if this program uses the 3741 as output.
NO	NO is the default value if the 3741 is not used by this program.

PGMDATA- { NO / YES }

The PGMDATA parameter specifies whether a program request from a terminal can have data entered with it. The default value if the parameter is not given is YES. If a terminal operator gives data with the program request, and the value in this parameter is NO, then the terminal operator receives an error message. A program specified as servicing multiple requesting terminals (parameter MRTMAX) must not specify PGMDATA-NO. If PGMDATA-YES is specified, the requesting terminal is not required to enter data. However, the program must be able to handle zero data length input.

If PGMDATA-NO is specified, any program request containing data is rejected by the CCP.

PGMDATA-YES must be specified for CCP/Disk Sort programs to do an accept input operation to free the console if it was the requesting device, and a release terminal operation if the requesting device was a terminal.

ENDMSG- { NO / YES }

The ENDMSG parameter specifies whether the terminal that requested this program will receive a message from the CCP when:

- The program goes to end-of-job
- The program releases the terminal

The default, if this parameter is omitted, is YES, indicating that a message is to be sent at these times. A specification ENDMSG-NO indicates that the following four messages are not to be sent to the requesting terminal:

S03	PROG END–PROCEED
S05	PROG END–SHUTDOWN
S01	PROG REL–PROCEED
S02	PROG REL–SHUTDOWN

Use of this parameter with the value NO might be particularly important if this program is requested by 3270 type terminals. If the display is to remain intact after the program releases the terminal or goes to end of job, ENDMSG-NO must be specified.

TERMS-'termname[/attrid] [,termname[/attrid]] . . . '

The TERMS parameter specifies terminals that are required to be available when this program is requested to execute. Terminal attributes sets can also be specified for these terminals.

The values for this parameter can be entered as split-values and/or sublists.

The value for TERMNAME is the symbolic name of a required terminal and must correspond to the NAME parameter on a TERMNAME statement. Each terminal name given for this parameter within one program must be unique. The terminal names do not have to be assigned to a specific terminal ID at assignment time. However, the system operator must assign them prior to the execution of this program. No two terminal names can be assigned to the same terminal ID.

The value for ATTRID specifies a terminal attributes set to be associated with this terminal during execution of this program. The value must be one of the attribute set values specified for this terminal on the BSCATERM or MLTATERM statement. If the attrid value is omitted, the first attributes set given on the BSCATERM or MLTATERM statement apply.

The requesting terminal is not normally specified as a required terminal, but if it is, the above conditions apply.

The maximum number of required terminals allowed within one PROGRAM statement is 80 or fewer, and the maximum number of specified files allowed within one PROGRAM statement is 40 or fewer, depending on the following formula:

$$T + 2F < 112$$

where

F= the number of filenames entered in the FILES parameter on this statement

T= the maximum number of required terminals allowed on this statement

FILES-'filename/access[/{SHR / NOSHR / KSORT}]

[,filename/access, . . .]'

The FILES parameter must be specified if disk files are used in this program. The values for each file are given as split-values and if multiple disk files are used, the entire value for the FILES parameter is a sublist.

The value for **filename** is required for each disk file used and must be a disk file name as specified in the program. Each filename specified here must correspond to the NAME parameter given on a DISKFILE statement or SYMFILE statement. See the formula under the TERMS parameter for the number of filenames that can be entered.

The value for **access** is required and describes the assumed organization and mode of access of this file as used by this application program. For additional discussion about the value for access, see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579.

The following terms can be entered for the access value:

CO–	Consecutive output
CG–	Consecutive input
CU–	Consecutive update
CA–	Consecutive add
DG–	Direct input
DU–	Direct input and update
DO [1] –	Direct output
IS–	Indexed sequential input only
ISA–	Indexed sequential input and add
ISL–	Indexed sequential input with limits
ISU–	Indexed sequential input and update
ISUL–	Indexed sequential input and update with limits
ISUA–	Indexed sequential input, update, and add
IA–	Indexed sequential add only
IR–	Indexed random input only
IRA–	Indexed random input and add
IRU–	Indexed random input and update
IRUA–	Indexed random input, update, and add
IO–	Ordered indexed load
IOU–	Unordered indexed load.

$$\left\{\begin{array}{l}\underline{\text{SHR}}\\ \text{NOSHR}\\ \text{KSORT}\end{array}\right\}$$

This optional parameter specifies whether this file can be shared with another program while this program is executing. The CCP allows the sharing of disk files in either input or update mode but the sharing of the same record at the same time is prohibited within the CCP. SHR, the default value for input, update, and add files indicates this file can be shared with another program. NOSHR, the default for load files, indicates this file is not shared with any other program during execution of this program. Files that are defined more than once within the same program cannot be opened as NOSHR. For Program Number 5704-SC1, SHR must not be specified if the access is CO, IO, IOU, CA, ISA, or ISUA. For Program Number 5704-SC2, SHR must not be specified if access is CO, IO, or IOU.

NOSHR must be specified for CCP/Disk Sort work files and output files.

[1] *Notes:* DO, direct output, allows the following:
1. The file can be a new file created at CCP startup.
2. The file is initialized to blanks at CCP startup. This eliminates the need for the file to be initialized before CCP startup.
3. The user can code his programs as DU (to allow checking for possible synonyms or updating records) but still specify DO in the // PROGRAM FILES keyword to allow steps 1 and 2 above.

KSORT (Program Number 5704-SC2)

This optional parameter specifies whether the keys added to an indexed file, during the execution of the task, should be sorted and merged into the existing file index. KSORT can be specified only if the access is IA, ISA, ISUA, IRA or IRUA. If KSORT is specified, the file cannot be shared.

$$\text{PACK-}\left\{\begin{array}{l}\underline{\text{PROGRAM}}^{1}\\ \text{SYSTEM}^{1}\end{array}\right\}$$

The PACK parameter specifies the disk pack that contains this program. PROGRAM is the default value and indicates this program resides on the pack $CCP was loaded from. SYSTEM indicates this program resides on the pack from which the user loaded DSM (IPL). PACK-R1, F1, R2, F2 or D1A through D4H (as noted above) indicates the logical unit that contains a program (Program Number 5704-SC2 only).

The PACK parameter specified for programs $CCPCL, $CCPOP, and $CCPDD is restricted to PROGRAM.

For EXECFIND-YES programs, the pack parameter specification is restricted to PACK-SYSTEM or PACK-PROGRAM (Program Number 5704-SC2 only). This allows CCP to use all 24 libraries to contain various user programs.

Placing programs or formats in a library designated by a physical unit may cause resident CCP to expand by 2K bytes.

PRUFLNG-nnn

The PRUFLNG parameter specifies the size of the program request under format (PRUF) input data coming from a 3270 screen. PRUF allows data with the program request to exceed the restriction of up to 78 characters normally associated with non-PRUF program requests.

[1] For Program Number 5704-SC2, the following codes may also be specified:

R1, F1, R2, F2
D1A, D1B, D1C, D1D
D2A, D2B, D2C, D2D
D3A, D3B, D3C, D3D
D3E, D3F, D3G, D3H
D4A, D4B, D4C, D4D
D4E, D4F, D4G, D4H

Specifying this keyword indicates that the PRUF option should be used during execution of this set. The PRUF option causes all text received with a 3270 program request to be passed to the program. (Without PRUF, only the first data field from the 3270 screen is passed to the user program as data with the program request.) If you specify the PRUFLNG parameter, you cannot specify PGMDATA-NO.

For non-DFF formats, the length you specify indicates the maximum number of text characters that can be received from the 3270 screen as a result of the program request. (This includes the 3270 text header and all 3270 field and control characters.)

For DFF formats, the PRUFLNG parameter specifies the maximum number of the text characters (not the size of the program's input record area) that can be received from the 3270 screen for the format specified for PRUF$Z. For more information, see the *IBM System/3 Communications Control Program Programmer's Reference Manual,* GC21-7579.

PRUF$Z-format name

The PRUF$Z parameter specifies the format name that should be used by the Display Format Facility (DFF) to de-code the data received with the program request. When this parameter is specified, the PRUFLNG, DFFMTERM, DFFSFDT, and DFFNDF parameters must also be specified on this statement. For more information, see the *IBM System/3 Communications Control Program Programmer's Reference Manual,* GC21-7579.

DFFMTERM-n

The DFFMTERM parameter specifies the number of terminals a program can be connected to; therefore, when MRT-programs are used this parameter must not be less than the MRTMAX-parameter.

A 1- to 3-digit value from 1 through 239 can be specified. If this parameter is omitted it is assumed none of the terminals serviced by this program will use DFF.

If this parameter is specified, the parameters DFFNDF and DFFSFDT must also be specified on this statement.

DFFNDF-n

The DFFNDF parameter specifies the maximum number of display format names that are referred to during any execution of this program.

A 1- to 3-digit value from 1 through 255 can be specified. If this parameter is omitted, it is assumed no terminals serviced by this program will use DFF.

If this parameter is specified, the parameters DFFMTERM and DFFSFDT must also be specified on this statement.

DFFSFDT-n

The DFFSFDT parameter specifies the size of the largest field descriptor table of any display format used by this program.

The value for this parameter can be from 1 through 3584 but must correspond to the largest field descriptor table used, as printed by the display format generation routine. If this parameter is omitted, it is assumed no terminals serviced by this program will use DFF.

If this parameter is given, the parameters DFFMTERM and DFFNDF must also be given on this statement.

MORCOR-n

The MORCOR parameter specifies additional 2K blocks of main storage allocation to the program for memory resident overlays (MROs) and external buffers. You can specify a one or two digit value from 1 through 99. If the additional storage is not available at program execution time, the program is executed without MROs. To use the MROs with the additional storage, the program must be link-edited with an MRO option.

SORT-{NO / YES}

This optional parameter (used for Program Number 5704-SC2 only) specifies whether this program is a CCP/Disk Sort program. When SORT-YES is specified, three files must be specified on the FILES parameter of the PROGRAM statement:

- An input file with consecutive input (CG) access. Note that CG access should be used on direct and indexed files (maximum of eight input files).
- A work file with consecutive add (CA) access and RECL-256 specified on the DISKFILE statement regardless of the record length of the input and output files (ORG-C on the DISKFILE assignment statement).
- An output file with consecutive output (CO) access (ORG-C on the DISKFILE assignment statement).

Duplicate filenames between the three file accesses are not allowed. These names must reference the sort files identified on the CCP/Disk Sort generation OCL file statements.

The NOSHR parameter must be specified for work files and output files. Sharing of input files is allowed; however, unpredictable results may occur due to updates entered by sharing tasks.

The PGMDATA-YES parameter must be specified so that the CCP/Disk Sort program will release the requesting terminal when the program is loaded.

EXECFIND-{NO / YES}

The EXECFIND parameter (for Program Number 5704-SC2 only) allows the specification of execution time program find (YES), or program find during startup (NO). The default is NO. Specifying YES forces CCP to locate and load the designated program every time it is called. Excessive use of EXECFIND can decrease performance.

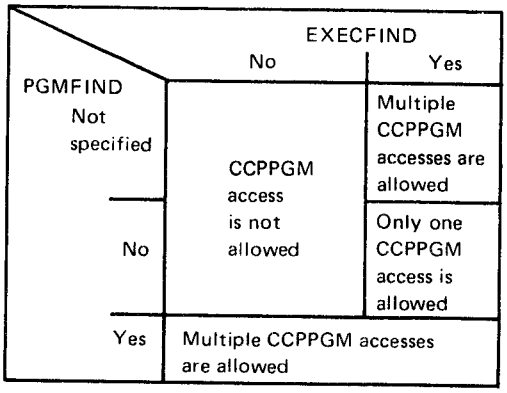
PACK-PROGRAM or PACK-SYSTEM must be specified for EXECFIND-YES programs.

PGMFIND-{YES / NO}

The PGMFIND parameter (for Program Number 5704-SC2 only) allows the location of the program to be changed by the program find routine (CCPPGM). Refer to the following chart. Specifying PGMFIND-YES allows the program find routine to access the program an unlimited number of times. Specifying PGMFIND-NO prohibits the program find routine from accessing the program, unless EXECFIND-YES is also specified. If PGMFIND-NO and EXECFIND-YES are both specified, the program find routine is allowed to access the program only one time.

If PGMFIND is not specified, the program find routine cannot access the program, unless EXECFIND-YES is specified. In this case, the program find routine has access to the program an unlimited number of times.

The effects of the PGMFIND parameter are different from the effects of the EXECFIND parameter. With PGMFIND, the system performance is not impaired, and no restriction is placed on the location of the application programs.



PRIORITY-{NORMAL / LOW}

The PRIORITY parameter (for Program Number 5704-SC2 only) specifies the priority a program is to have as a CCP task. The default is NORMAL.

TASKSIZE-n (Program Number 5704-SC2 only)

The TASKSIZE parameter specifies the main storage size (in 1K blocks) to be allocated by the program. This size overrides the program size (in sectors) contained within the header record of the object program. You may specify a value from 4 to 32. When TASKSIZE is specified, EXECFIND-YES cannot be specified.

PROGRAM FIND ROUTINE (PROGRAM NUMBER 5704-SC2 ONLY)

The program find routine (CCPPGM) allows a user to access a new or revised version of a program (on the same or different unit) that was specified in the assignment set. The use of CCPPGM is governed by the EXECFIND and PGMFIND parameters on the // PROGRAM statement.

The program find routine is executed under CCP and may be requested from the console or from any command capable terminal.

If EXECFIND-YES had been specified on the // PROGRAM statement, the program find routine can be used to change that status to EXECFIND-NO, thereby decreasing the time required to fetch that program.

EXECFIND programs must reside on the CCP program or system pack, whereas PFMFIND programs may reside in any simulation area library. The program find routine can be applied to all versions or copies of a program in any simulation area to which CCP has access, assuming that the program had been defined in the assignment set. (CCP has access to the system pack, program pack, F1, R1, F2, and R2, as well as to any physical simulation area that contains application programs or DFF formats that are currently being used.)

The use of CCPPGM may cause the assignment set being executed and its assignment set source to be inconsistent. When programs are found, the assignment set source should be updated to reflect the revised status of the program, and updates should be made to the PACK parameter if necessary.

To use CCPPGM, make the following entries in the assignment set:

```
// TERMATTR ATTRID-04,DATAFORM-MESSAGE,
   DFF3270-NO
// PROGRAM NAME-CCPPGM,NEVEREND-YES,
   ENDMSG-NO
```

For additional information on the use of CCPPGM, refer to the *IBM System/3 Model 15 Communications Control Program System Operator's Guide*, GC21-7619.

Note: When CCPPGM is included in an assignment set, ATTRID-04 is required for Program Find but may also be used for FORMAT FIND (CCPFMT).

FORMAT FIND ROUTINE (PROGRAM NUMBER 5704-SC2 ONLY)

The format find routine (CCPFMT) makes formats that have been created or modified by DFGR or PFGR in another partition, available to programs in the current assignment set. The routine is executed under CCP and performs the following functions:

- Places the current object library C/S (cylinder/sector) location of the format into the format index in $CCPFILE.
- Updates the DFFSFDT value of the PROGRAM assignment statement.

To use CCPFMT, make the following entries in the assignment set:

```
// TERMATTR ATTRID-04,DATAFORM-MESSAGE,
   DFF3270-NO
// PROGRAM NAME-CCPFMT,NEVEREND-YES,
   ENDMSG-NO
```

The routine is invoked from either the system operator's console or a command terminal. For additional information on the use of CCPFMT, refer to the *Model 15 CCP System Operator's Guide*, GC21-7619.

Note: When CCPFMT is included in an assignment set, ATTRID-04 is required for format find.

DISK FILE CLOSE/OPEN FACILITY

This facility allows the system operator to dynamically close and re-open disk files opened at startup. By doing this, you can merge added records with existing records in add files. By closing and re-opening an indexed add file you can:

- Improve performance for indexed random add accesses
- Add sequentially to an indexed file more than once per execution of CCP

Note: Running $CCPCL will not allow batch users to use SHARE-NO on any file that opened at CCP startup.

- Retrieve added records without including add data management in CCP programs
- Run programs in the non-CCP partition that access the added records in the closed files for retrieval or update

Records loaded to a CCP output file remain unaccessable to other partitions until CCP is shutdown.

When run on a multivolume file, the Close/Open facility will close or open the entire file while it is being used in the current assignment set.

To use the Disk File Close/Open facility, you must specify the following programs in the assignment set: $CCPCL and $CCPOP. For a description of the required assignment statements, see the keywords NAME and MRTMAX of the PROGRAM statement.

If the indexed add files will be closed, the following OCL statements should be included in the CCP startup OCL to reserve space for the following files:

// FILE NAME-$INDEX44 (for indexed add files on 5444 drives, Program Number 5704-SC1 only)

// FILE NAME-{$INDEX40 / $INDEX45} (for indexed add files on 5445, 3340, or 3344 drives)

// FILE NAME-$INDEX40 (for indexed add files on 3340 or 3344 drives, Program Number 5704-SC2 only)

ASSIGNMENT LIST PROGRAM

The assignment list program ($CCPAL)[1] can be executed after the assignment file build program ($CC1BF) is executed. $CCPAL has four purposes:

1. List either the contents of all the assignment sets in the assignment file, $CCPFILE, or the contents of any individual set in the file, to show the contents of the set or sets.

2. List the request count for each program in the assignment file, if the program request count option was chosen at generation time. The request counts can be listed either separately for each set or as a total for all sets. The user can clear the request count to 0 either after listing the counts or without listing the counts.

3. List the assignment file directory.

4. List the CCP configuration. This is allowed anytime after generation, even before an assignment run.

The CCP assignment list program ($CCPAL) can be used to list the following assignment file data:

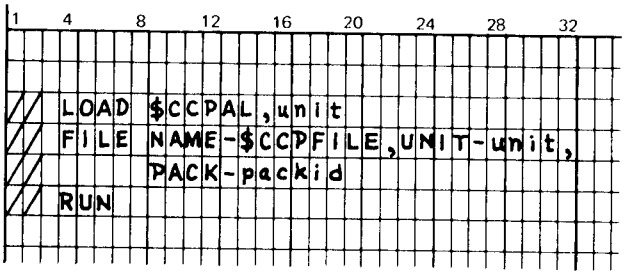
- Assignment file configuration record, directory, or assignment sets.

- An option to list and/or reset the program-request count (if request count was selected at generation).

The input for the assignment list program is as follows:

- The previously built file $CCPFILE.

- The OCL statement provided by the user. The source statements must be input from the system input device. The following OCL statements are needed:



- The assignment control statements instructing $CCPAL of the function(s) it is to perform.

  A combination of parameters on the // LIST control statement to define the options the user wishes to exercise in listing the file, $CCPFILE.

- The /* statement at the end of all input control statements

The output for the assignment list program is as follows:

- A user selected listing of portions of the file $CCPFILE.

- A user selected resetting of the program-request count within the file (if request count was selected at generation).

All listings are output to the printer.

Keyword parameters are listed in the form used by the assignment build program control statements. The date and time headers of the system information table (SIT) denote the date and time of the assignment build and do not represent valid assignment keywords.

[1] For more information concerning the execution of $$CCPAL, see *Assignment Restrictions* in this chapter (5704-SC2 only).

// LIST Statement

The assignment list program is controlled by the list statement, which the user enters from the system input device. The list statement specifies:

- What is to be listed—an individual set, all sets, only the control file directory, or only the CCP configuration.
- Whether the program-request count should be printed for one or all of the sets in the file.
- Whether the program-request count should be reset to zero for one or all of the sets in the file.

```
// LIST    [SET- { id     } ] [,PGMSTAT- { NO  } ]
                 { ALL    }              { YES }
                 { DIR    }
                 { CONFIG }
            [,RESETPS- { NO  } ]
                       { YES }
```

The statement identifier is the word *list.* Any number of list statements can be entered for a single execution of $CCPAL. No continuation statements are allowed. This control statement must be given but no parameters are required on the statement.

If no parameters are specified, the statement contains the following information only:

```
// LIST
```

All defaults are assumed, the contents of all the sets in the file are listed, and the program-request count is left untouched.

SET- { ID   }
     { ALL   }
     { DIR   }
     { CONFIG }

The SET parameter specifies which parts of the $CCPFILE to list.

The value *id* refers to any valid assignment set that exists in the control file and causes that set to be printed provided no other parameters are included in this statement. If the parameters PGMSTAT and/or RESETPS are included, the *id* value refers to all request counts within that set.

The value *ALL*, the default value, causes all assignment sets to be listed provided no other parameters are included in the statement. If the parameter PGMSTAT and/or RESETPS is included, the value *ALL* refers to all request counts within all assignment sets in the file.

The value *DIR* causes the control file directory (contains the IDs of assignment sets contained in the file) to be printed. If the value SET-DIR is given, no other parameters can be given on this statement.

The value *CONFIG* indicates the CCP configuration record should be printed. If the value SET-CONFIG is given, no other parameters can be given on this statement.

PGMSTAT- { NO }
         { YES }

The PGMSTAT parameter states whether the program-request counts should be printed. The program-request counts for all assignment sets are printed if the value for the SET parameter is ALL or not given, the value YES is given for this parameter, and the option was chosen at generation time. If the value given is for a particular valid set ID and the value YES is given for this parameter, the request counts for just that set are printed. The default, if this parameter is omitted, is NO. If the value YES is given for this parameter, and the system was defined at generation time as not containing request counts, a diagnostic message is given and this statement is ignored.

Note: If an assignment build is run to replace the set CCP is executing from, the program count will be reset to zero for that set. If an assignment build is run to delete the set CCP is executing from, the program count is lost for that set.

RESETPS- {NO / YES}

The RESETPS parameter states whether the program-request counts should be reset to zero. If the value YES is given and the value *ALL* is given (or defaulted) for the SET parameter on this statement, the program-request counts for all sets in the file are reset to zero.

If the value YES is given and a valid set ID is given for the SET parameter on this statement, the program-request counts are reset to zero for just that set. If the program request count is reset for the set CCP is executing from, the program request count will be lost for that execution. The default, for the omitted parameter is NO. If the value YES is given with this parameter, and the system was defined at generation time as not containing request counts, a diagnostic message is given and this statement is ignored.

Combinations of parameters and their associative meaning for the LIST statement follows. Whenever a parameter is omitted, the same effect would occur if the parameter were given with its default value.

// LIST
List all sets in the file.

// LIST SET-id
List this particular set.

// LIST SET-ALL
List all sets in the file.

// LIST SET-DIR
List only the directory.

// LIST SET-CONFIG
List only the configuration record.

// LIST PGMSTAT-YES
List the program request-counts for all sets.

// LIST RESETPS-YES
Reset the program request-counts to zero for all sets.

// LIST SET-id,PGMSTAT-YES
List the program request-counts for this set only.

// LIST SET-id,RESETPS-YES
Reset the program request-counts for this set only.

// LIST PGMSTAT-YES,RESETPS-YES
List the program request-counts for all sets and reset them to zero.

// LIST SET-id,PGMSTAT-YES,RESETPS-YES
List the program request-counts for this set and reset them to zero after listing them.

SAMPLE ASSIGNMENT BUILD EXECUTION

As indicated in the sample assignment set for card-oriented assignment, the dummy 1 and dummy 2 program, and the DISKFILE statements should be deleted before running this assignment set.

```
// LOG PRINTER                                                          00070000
// LOAD $CCPAS,R1                                                       00080000
*                                                                       00090000
*****FILL IN PACK AND UNIT                                             00100000
*                                                                       00110000
// FILE NAME-$CCPFILE,RETAIN-P,UNIT-R1,PACK-R1R1R1                      00120000
*                                                                       00130000
*****FILL IN PACK AND UNIT                                             00140000
*                                                                       00150000
// FILE NAME-$CCPWORK,RETAIN-S,TRACKS-3,UNIT-R1,PACK-R1R1R1             00160000
// RUN
```

```
*                                                                       00170000
*********THE FOLLOWING STATEMENTS CAN BE MODIFIED FOR YOUR              00180000
*********CONFIGURATION BUT SOME MUST BE KEPT TO RUN CCPIVP.  SEE THE    00190000
*********COMMENTS IN THIS DECK.                                         00200000
*                                                                       00210000
// SET ID-a,ACTION-CREATE,DFLTEXEC-YES                                  00220000
// SYSTEM MINUPA-16K,MINTPBUF-3090,                                     00230000
//       PASSWORD-FECD,                                                 00240000
//       COMMANDL-50,DFFPACK-PROGRAM,PGMREQL-15                         00250000
*                                                                       00260000
// TERMATTR ATTRID-1,TRANSLAT-NO,BLKL-512,DATAFORM-MESSAGE,             00270000
//       VERIFYIC-NO,DFF3270-YES                                        00280000
*                                                                       00290000
*********THIS STMNT TYPE REQD FOR CCPIVP(OR MLTALINE STMNT)             00300000
*                                                                       00310000
// BSCALINE TYPE-CS,LINENUM-1,POLLLIST-'00,01,10,11'                    00320000
// BSCATERM TERMID-00,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,    00330000
//       ADDRCHAR-*60604040*,POLLCHAR-*40404040*                        00340000
// BSCATERM TERMID-01,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,    00350000
//       ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*                        00360000
// BSCATERM TERMID-10,TYPE-3277M2,ATTRID-1,COMMAND-NO,                  00370000
//       ADDRCHAR-*61614040*,POLLCHAR-*C1C14040*                        00380000
// BSCATERM TERMID-11,TYPE-3277M2,ATTRID-1,COMMAND-NO,                  00390000
//       ADDRCHAR-*6161C1C1*,POLLCHAR-*C1C1C1C1*                        00400000
*                                                                       00410000
// TERMNAME NAME-CU0DV0,TERMID-00                                       00420000
// TERMNAME NAME-CU0DV1,TERMID-01                                       00430000
// TERMNAME NAME-CU1DV0,TERMID-10                                       00440000
// TERMNAME NAME-CU1DV1,TERMID-11                                       00450000
*                                                                       00460000
*********THIS STMNT TYPE REQD FOR CCPIVP                                00470000
*                                                                       00480000
// DISKFILE NAME-CGIVFIL1,ORG-C,RECL-16                                 00490000
*                                                                       00500000
*********THIS STMNT TYPE REQD FOR CCPIVP                                00510000
*                                                                       00520000
// DISKFILE NAME-CGIVFIL2,ORG-C,RECL-16                                 00530000
*                                                                       00540000
*********NOTE THAT ONE DISKFILE STATEMENT -CGIVFILE- WOULD BE NEEDED    00550000
*********IF SYMBOLIC FILES ARE NOT BEING USED.                          00560000
*                                                                       00570000
* THE FOLLOWING TWO DISK FILE STATEMENTS ARE FOR EXAMPLE ONLY AND       00580000
* SHOULD BE REMOVED PRIOR TO RUNNING THIS ASSIGNMENT SET.               00590000
*                                                                       00600000
// DISKFILE NAME-DUMMY1,ORG-D,RECL-256                                  00610000
// DISKFILE NAME-DUMMY2,ORG-I,RECL-64,KEYL-8,KEYPOS-1,MSTRINDX-YES[1]   00620000
*                                                                       00630000
*********THIS STMNT TYPE REQD FOR CCPIVP IF SYMBOLIC FILES ARE USED.    00640000
*                                                                       00650000
// SYMFILE NAME-CGIVFILE,DISKFILE-'CGIVFIL1,CGIVFIL2'                   00660000
*                                                                       00670000
*************THIS STMNT NECESSARY FOR CCPIVP,PACK AND PRINTER VALUES    00680000
*************CAN BE CHANGED FOR YOUR CONFIG.                            00690000
*                                                                       00700000
// PROGRAM NAME-CCPIVP,PGMDATA-YES,                                     00710000
//       FILES-'CGIVFILE/CO/NOSHR',                                     00720000
//       PACK-PROGRAM,PRINTER-NO                                        00730000
*                                                                       00740000
*********NOTE THAT CCPIVP MUST BE ON CORRECT PACK AT STARTUP OF CCP.    00750000
*                                                                       00760000
```

[1] Not applicable for Program Number 5704-SC2.

```
* THE FOLLOWING TWO PROGRAM STATEMENTS ARE FOR EXAMPLE ONLY AND          00770000
* SHOULD BE REMOVED PRIOR TO RUNNING THIS ASSIGNMENT SET.               00780000
*                                                                       00790000
// PROGRAM NAME-DUMMY1,MRTMAX-2,PGMDATA-YES,                            00800000
// FILES-'DUMMY1/DU/SHR,DUMMY2/IRUA/SHR',PACK-SYSTEM,DFFMTERM-4,        00810000
// DFFNDF-2,DFFSFDT-1150                                                00820000
*                                                                       00830000
// PROGRAM NAME-DUMMY2,MRTMAX-2,PGMDATA-YES,                            00840000
// FILES-'DUMMY1/DU/SHR,DUMMY2/IRA/SHR',PACK-SYSTEM,DFFMTERM-2,         00850000
// DFFNDF-1,DFFSFDT-396                                                 00860000
*                                                                       00870000
*                                                                       00880000
                                                                        00890000

0       WARNING MESSAGES

0       TERMINATION MESSAGES
```

For Program Number 5704-SC2

```
                    $CCPFILE CONFIGURATION              00/00/00

SIGN-ON SECURITY - CCP
PROGRAMMING LANGUAGE SUPPORT  COBOL  FORTRAN  ASSEMBLER  RPGII
PROGRAM REQUEST-COUNT STATISTICS KEPT
DATA MODE ESCAPE CHARACTERS  //////
3270 DISPLAY FORMATTING

SICC PORT LINE SUPPORT

TASK-TO-TASK FACILITY SUPPORT

NUMBER OF BSCA LINES AVAILABLE TO CCP  2
BSCA LINE-TYPE SUPPORT  POINT-TO-POINT  MULTIPOINT  CONTROL-STATION
     SWITCHED  PORTLINE
BSCA FACILITIES                    RECORD-SEPARATOR-X¤1E¤  GET-MESSAGE
     EBCDIC  ASCII  AUTO-RESPONSE  RESIDENT-POLLING  INTERVAL-POLLING
     BSCA-TRANSP  PRUF
BSCA DEVICE-TYPE SUPPORT
     3275M1
     3277M1
     3284M1
     3286M1
     3275M2
     3277M2
     3284M2
     3286M2
     3735
     CPU
     3741

NUMBER OF BSCC LINES AVAILABLE TO CCP  2
BSCC LINE-TYPE SUPPORT  CONTROL-STATION  POINT-TO-POINT  PORTLINE
BSCC FACILITIES                    RECORD-SEPARATOR-X¤1E¤  GET-MESSAGE
     EBCDIC  ASCII  INTERVAL-POLLING
BSCC DEVICE-TYPE SUPPORT
     3275M1
     3277M1
     3284M1
     3286M1
     3275M2
     3277M2
     3284M2
     3286M2
     3735
     CPU
     3741
```

For Program Number 5704-SC2

```
              $CCPFILE DIRECTORY LIST                   00/00/00

NUMBER OF SETS IN THIS CCP FILE                                1
NUMBER OF ENTRIES AVAILABLE FOR SETS                          24

EXECUTION TIME DEFAULT SET ID  WITH NO STARTUP PROMPTS       *@*
DATE FILE LAST UPDATED                                  00/00/00

SET ID IN THE FILE                    LENGTH OF SET IN SECTORS
        @                                               7
```

For Program Number 5704-SC1:

```
SET-a                          $CCPFILE SYSTEM INFORMATION TABLE                    00/00/00

        MINTPBUF        POLTIME     PRINTER      RP1442    COMMANDL  DFFPACK          DATE    TIME
MINUPA          PASSWORD       N3741        MFCU       R2501   PGMLNG          FSQE

 18K     3090   FECD     60,2   NO   NO     NO    NO    N   50   15 PROGRAM      05/01/76 04 36
```

For Program Number 5704-SC2:

```
SET-a                          $CCPFILE SYSTEM INFORMATION TABLE                    04/07/80

MINUPA  MINTPBUF  PASSWORD  POLTIME  PRINTER  MFCU  RP1442  R2501  COMMANDL  PGMLNG  DFFPACK  DFFINDX
 22K     2840     FECD      60,2     NO       NO    NC      NO     NO        50      15       PROGRAM  NO

  DATE          MAXCHAIN     ERTIME      DUMPUNIT
        TIME          CSPOLTIM   PRUFOF
04/07/80 00 22      0        60,2  05    NO

SET-a                          $CCPFILE TERMINAL ATTRIBUTES TABLE

ATTRID       UPCASE      DATAFORM                BLKL          TRANSP          SPAN          DFF3270
      TRANSLAT     SWITCHED              RECL              ITB          VERIFYID      VARL

    1   NO     NO     NO     MESSAGE              512   NO     NO     NO     NO     NO     YES
```

For Program Number 5704-SC1:

```
SET-a                          $CCPFILE LINE CONTROL TABLE                          00/00/00

LINENUM      DATARATE       RCVINT       DBLBUF              NRETRY  CALC BSCA          WAIT
        TYPE          AUTOPOLL      TIOLT          XMCODE             BLOCK SIZE  RECL

BSCA-1   CS                                  NO     EBCDIC       7       554            180

    LINENUM  DELAY/     IDEXSEND                        POLLLIST
             POLLLOOP

    BSCA-1     18                                       00,01,10,11
```

For Program Number 5704-SC2:

```
SET-a                          $CCPFILE LINE CONTROL TABLE                          00/00/00

LINENUM      DATARATE       RCVINT       DBLBUF              NRETRY  CALCULATED          WAIT
        TYPE          AUTOPOLL      TIOLT          XMCODE             BLOCK SIZE  RECL

BSCA-1   CS                                  NO     EBCDIC       7       554            180

    LINENUM  DELAY/     IDEXSEND                        POLLLIST        DFFBUF
             POLLLOOP

    BSCA-1     5                                        00,01,10,11     NO
```

For Program Number 5704-SC1:

```
SET-a                          $CCPFILE TERMINAL USED TABLE

LINE        TYPE                    ONLINE       ATTRID  PINCOMP  ADDRCHAR          POLLCHAR
    TERMID                  COMMAND      OFFACTN      ADDR  POUTCOMP  IDEXRCV

B1   00   3277/84/86-M2   YES   YES   HOLD      1            60604040          40404040
B1   01   3277/84/86-M2   YES   YES   HOLD      1            6060C1C1          4040C1C1
B1   10   3277/84/86-M2   NO    YES             1            61614040          C1C14040
B1   11   3277/84/86-M2   NO    YES             1            6161C1C1          C1C1C1C1
```

For Program Number 5704-SC2:

```
SET-@                          $CCPFILE TERMINAL USED TABLE                          CC/00/00

LINE   TERMID   TYPE              COMMAND   ONLINE   OFFACTN   ATTRID   PINCOMP   ADDRCHAR   POLLCHAR   AUTO
                                                                ADDR     PCUTCCMP  IDEXRCV               ERP
B1     00    3277/84/86-M2        YES       YES      HCLD      1                  60604040   40404040   NO
B1     01    3277/84/86-M2        YES       YES      HCLD      1                  6060C1C1   4040C1C1   NO
B1     10    3277/84/86-M2        NO        YES                1                  61614040   C1C14040   NO
B1     11    3277/84/86-M2        NO        YES                1                  6161C1C1   C1C1C1C1   NO

SET-@          $CCPFILE TERMINAL NAME TABLE

 NAME      TERMID    MSTRNAME   INCOMP   OUTCOMP  ID  PHONENUM

CU0DV0       00
CU0DV1       01
CU1DV0       10
CU1DV1       11
```

For Program Number 5704-SC1:

```
SET-@                          $CCPFILE FILE CONTROL TABLE

FILENAME      ORG      OPENED       RECL      KEYL     KEYPOS     MSTRINDX    MIXSIZ

CGIVFIL1       C       OUTPUT         16
CGIVFIL2       C       OUTPUT         16
DUMMY1         D       UPDATE        256
DUMMY2         I       ADD            64        8          1         YES           0

SYMBOLIC NAME  DISKFILE NAME/S

CGIVFILE       CGIVFIL1  CGIVFIL2
```

For Program Number 5704-SC2:

```
SET-@                          $CCPFILE FILE CONTROL TABLE                          CC/00/00

FILENAME      ORG      OPENED     VOLUMES    RECL      KEYL     KEYPOS

CGIVFIL1       C       OUTPUT                 16
CGIVFIL2       C       OUTPUT                 16
DUMMY1         D       UPDATE                256
DUMMY2         I       ADD                    64        8          1

SYMBOLIC NAME  DISKFILE NAME/S

CGIVFILE       CGIVFIL1  CGIVFIL2
```

For Program Number 5704-SC1:

```
SET-@                                $CCPFILE PROGRAM CONTROL TABLE                                00/00/00

PROG          NEVEREND    MFCU/M      R2501       PGMDATA    DFFSFDT      PACK  PROG[1]
NAME   MRTMAX      N3741      PRINTER     RP1442     ENDMSG       DFFWKA     SIZE     PRUF$Z

CCPIVP            NO NO     NO     NO     NO     NO     YES  YES                    P
DUMMY1     2      NO NO     NO     NO     NO     NO     YES  YES   1280  1792     S
DUMMY2     2      NO NO     NO     NO     NO     NO     YES  YES    512   768     S

    PROG     FILES                          TERMS                  MORCOR
    NAME         FILENAME   ACCESS     SHR      TERMNAME  ATTRID

    CCPIVP       CGIVFILE   LOAD       NO       SORT
    DUMMY1       DUMMY1     UPDATE     YES
                 DUMMY2     ADD        YES
    DUMMY2       DUMMY1     UPDATE     YES
                 DUMMY2     ADD        YES
```

[1]Program size is not printed by assignment list ($CCPAL) unless Startup has been executed on the assignment set.

For Program Number 5704-SC2:

```
SET-a                          $CCPFILE PROGRAM CONTROL TABLE                    04/07/80

PROG          NEVEREND    MFCU/M      R2501      PGMDATA    DFFSFDT      PACK PROG
NAME   MRTMAX      N3741     PRINTER    RP1442     ENDMSG     DFFWKA       SIZE      PRUF$Z

$CCPDD            NO   NO    NO    YES   NO    NO    YES  YES                    P
CCPIVP            NO   NO    NO    NO    NO    NO    YES  YES                    P
DUMMY1     2      NO   NO    NO    NO    NO    NO    YES  YES   1024  1536     S
DUMMY2     2      NO   NO    NO    NO    NO    NO    YES  YES    512   768     S

PROG   EXECFIND        SORT
NAME          PGMFIND       PRIORITY

$CCPDD    YES   NO        NO   NORMAL
CCPIVP    NO    NO        NO   NORMAL
DUMMY1    NO    NO        NO   NORMAL
DUMMY2    NO    NO        NO   NORMAL

     PROG      FILES                                          TERMS              MORCOR
     NAME           FILENAME   ACCESS      SHR   KSORT       TERMNAME  ATTRID
     CCPIVP         CCIVFILE    LOAD       NO    NO
     DUMMY1         DUMMY1     UPDATE      YES   NO
                    DUMMY2      ADD        YES   NO
     DUMMY2         DUMMY1     UPDATE      YES   NO
                    DUMMY2      ADD        YES   NO
```

SAMPLE ASSIGNMENT SET: CALCULATION OF MAIN STORAGE SIZES

```
// LOG PRINTER
*******SAMPLE ASSIGNMENT AND SAMPLE START-UP DECK********************** 00010000
*                                                                       00020000
*********SAMPLE ASSIGNMENT SET ************************************** 00030000
*                                                                       00040000
*****FILL IN UNIT                                                       00050000
*                                                                       00060000
// LOAD $CCPAS,R2                                                       00070000
*                                                                       00080000
*****FILL IN PACK AND UNIT                                             00090000
*                                                                       00100000
// FILE NAME-$CCPFILE,RETAIN-P,UNIT-R2,PACK-CCPOBJ                      00110000
*                                                                       00120000
*****FILL IN PACK AND UNIT                                             00130000
*                                                                       00140000
// FILE NAME-$CCPWORK,RETAIN-S,TRACKS-3,UNIT-R2,PACK-CCPOBJ             00150000
// RUN                                                                  00160000
*                                                                       00170000


$CCPFILE ASSIGNMENT BUILD LISTING              PAGE 001                 00/00/00


*********THE FOLLOWING STATEMENTS CAN BE MODIFIED FOR YOUR              00180000
*********CONFIGURATION BUT SOME MUST BE KEPT TO RUN CCPIVP.  SEE THE    00190000
*********COMMENTS IN THIS DECK.                                         00200000
*                                                                       00210000
// SET ID-a,ACTION-CREATE,DFLTEXEC-YES                                  00220000
// SYSTEM MINUPA-18K,MINTPBUF-3090,                                     00230000
//      PASSWORD-FECD, 10                  13                           00240000
//    2 COMMANDL-50,DFFPACK-PROGRAM,PGMREQL-15  1                       00250000
*                                                                       00260000
// TERMATTR ATTRID-1,TRANSLAT-NO,BLKL-512,DATAFORM-MESSAGE,             00270000
//      VERIFYID-NO,DFF3270-YES  11  12                                 00280000
*                                                                       00290000
*********THIS STMNT TYPE REQD FOR CCPIVP OR MLTALINE STMNT              00300000
*                                                                       00310000
// BSCALINE TYPE-CS,LINENUM-1,POLLLIST-¤00,01,10,11¤                    00320000
// BSCATERM TERMID-00,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,    00330000
//      ADDRCHAR-*60604040*,POLLCHAR-*4040404O*                         00340000
// BSCATERM TERMID-01,TYPE-3277M2,ATTRID-1,COMMAND-YES,OFFACTN-HOLD,    00350000
//      ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*                         00360000
// BSCATERM TERMID-10,TYPE-3277M2,ATTRID-1,COMMAND-NO,                  00370000
```

```
//       ADDRCHAR-*61614040*,POLLCHAR-*C1C14040*                    00380000
// BSCATERM TERMID-11,TYPE-3277M2,ATTRID-1,COMMAND-NO,              00390000
//       ADDRCHAR-*6161C1C1*,POLLCHAR-*C1C1C1C1*                    00400000
*                                                                   00410000
// TERMNAME NAME-CU0DV0,TERMID-00                                   00420000
// TERMNAME NAME-CU0DV1,TERMID-01                                   00430000
// TERMNAME NAME-CU1DV0,TERMID-10                                   00440000
// TERMNAME NAME-CU1DV1,TERMID-11                                   00450000
*                                                                   00460000
*********THIS STMNT TYPE REQD FOR CCPIVP                           00470000
*                                                                   00480000
// DISKFILE NAME-CGIVFIL1,ORG-C,RECL-16                             00490000
*                                                                   00500000
*********THIS STMNT TYPE REQD FOR CCPIVP                           00510000
*                                                                   00520000
// DISKFILE NAME-CGIVFIL2,ORG-C,RECL-16                             00530000
*                                                                   00540000
*********NOTE THAT ONE DISKFILE STATEMENT -CGIVFILE- WOULD BE NEEDED 00550000
*********IF SYMBOLIC FILES ARE NOT BEING USED.                     00560000
*                                                                   00570000
* THE FOLLOWING TWO DISK FILE STATEMENTS ARE FOR EXAMPLE ONLY AND   00580000
* SHOULD BE REMOVED PRIOR TO RUNNING THIS ASSIGNMENT SET.           00590000
*                                                                   00600000
// DISKFILE NAME-DUMMY1,ORG-D,RECL-256                              00610000
// DISKFILE NAME-DUMMY2,ORG-I,RECL-64,KEYL-8,KEYPOS-1,MSTRINDX-YES[1] 00620000
*                                                                   00630000
*********THIS STMNT TYPE REQD FOR CCPIVP IF SYMBOLIC FILES ARE USED. 00640000
*                                                                   00650000
// SYMFILE NAME-CGIVFILE,DISKFILE-¤CGIVFIL1,CGIVFIL2¤               00660000
*                                                                   00670000
*************THIS STMNT NECESSARY FOR CCPIVP,PACK AND PRINTER VALUES 00680000
*************CAN BE CHANGED FOR YOUR CONFIG.                        00690000
*                                                                   00700000
// PROGRAM NAME-CCPIVP,PGMDATA-YES,                                 00710000
//       FILES-¤CGIVFILE/CO/NOSHR¤,                                 00720000
//       PACK-PROGRAM,PRINTER-NO                                    00730000
*                                                                   00740000
```

```
$CCPFILE ASSIGNMENT BUILD LISTING          PAGE 002        00/00/00

*********NOTE THAT CCPIVP MUST BE ON CORRECT PACK AT STARTUP OF CCP. 00750000
*                                                                   00760000
* THE FOLLOWING TWO PROGRAM STATEMENTS ARE FOR EXAMPLE ONLY AND     00770000
* SHOULD BE REMOVED PRIOR TO RUNNING THIS ASSIGNMENT SET.           00780000
*                                                                   00790000
// PROGRAM NAME-DUMMY1,MRTMAX-2,PGMDATA-YES,                        00800000
// FILES-¤DUMMY1/DU/SHR,DUMMY2/IRUA/SHR¤,PACK-SYSTEM,DFFMTERM-4,    00810000
// DFFNDF-2,DFFSFDT-1150                                            00820000
*                                                                   00830000
// PROGRAM NAME-DUMMY2,MRTMAX-2,PGMDATA-YES,                        00840000
// FILES-¤DUMMY1/DU/SHR,DUMMY2/IRA/SHR¤,PACK-SYSTEM,DFFMTERM-2,     00850000
// DFFNDF-1,DFFSFDT-396                                             00860000
*                                                                   00870000
/*                                                                  00880000

0       WARNING MESSAGES

0       TERMINATION MESSAGES
```

7 3 5 8 6 4

[1] Not applicable for Program Number 5704-SC2.

1 Program request length is equal to length of program name + 1 blank + program data for non-PRUF program requests. For example:

CCPIVP
Program name = 6
Blank = 1
Data = 8
Total = 15

2 COMMANDL assures that 50 character command length is permitted.

3 DUMMY1 communicates with two requesting terminals and when loaded, issues acquires to two additional terminals. While DUMMY1 may communicate with only one of the two requesting terminals at a time, it can run with any or all of the four terminals at the same time. (Reference **9**)

4 DUMMY2 communicates with up to two requesting terminals at any one time. (Reference **9**)

5 1150 is the largest field descriptor table used by this program (DUMMY1 uses format $ZOREN and $Z0009). (Reference **9**)

6 396 is the size of the field descriptor table for the only format used by this program (DUMMY2 uses only format $Z0009). (Reference **9**)

7 DUMMY1 uses two data formats. (Reference **9**)

8 DUMMY2 uses one data format. (Reference **9**)

Use **3** **5** **7** in the formula:

DFF program appended storage (PAS) size equals:
(DFF PAS base size is 124 bytes.)

$$[\overbrace{(124 + (DFFMTERM \times 37) + (DFFNDF \times 18))}^{*}] + \overbrace{DFFSFDT}^{*}$$

For example, the DFF PAS size for DUMMY1:

$$= [\overbrace{(124 + (4 \times 37) + (2 \times 18))}^{*}] + \overbrace{[1150]}^{*}$$

= 124 + 148 + 36 (rounded to next highest 256) + 1280
= 308 (rounded to next highest 256) + 1280
= 512 + 1280 = 1792

Refer to the following printout of a $CCPDF run.

* Rounded to next highest 256.

$ZOREN DISPLAY FORMAT INFORMATION

EXECUTION TIME DATA - OUTPUT AREA FORMAT - END POSITIONS FOR RPG PROGRAMS

FIELD NAME	FIELD LENGTH	END POSITION	FIELD NAME	FIELD LENGTH	END POSITION	FIELD NAME	FIELD LENGTH	END POSITION
OPCODE	004	0004	LENGTH	004	0008	TMNAME	006	0014
$ZOREN	006	0020						

INPUT AREA FORMAT - END POSITIONS FOR RPG PROGRAMS

FIELD NAME	FIELD LENGTH	END POSITION	FIELD NAME	FIELD LENGTH	END POSITION	FIELD NAME	FIELD LENGTH	END POSITION
RTCODE	004	0004	LENGTH	004	0008	TMNAME	006	0014
- AID-	001	0015	CUSNO	006	0021	SNAME	022	0043
SADDR1	022	0065	SADDR2	022	0087	SADDR3	022	0109
QTY1	004	0113	ITEMN1	008	0121	QTY2	004	0125
ITEMN2	008	0133	QTY3	004	0137	ITEMN3	008	0145
QTY4	004	0149	ITEMN4	008	0157	QTY5	004	0161
ITEMN5	008	0169	QTY6	004	0173	ITEMN6	008	0181
QTY7	004	0185	ITEMN7	008	0193			

LENGTH OF OUTPUT RECORD AREA REQUIRED IN DFF PROGRAM
RPG *SUBR92* - 0020 OTHER - 0012

LENGTH OF INPUT RECORD AREA REQUIRED IN DFF PROGRAM

RPG *SUBR92* - 0193 OTHER - 0185

INFORMATION FOR USE DURING CCP ASSIGNMENT STAGE

THE DECIMAL LENGTH OF THE FIELD DESCRIPTOR TABLE IS 1150
THE DECIMAL LENGTH OF THE OUTPUT TEXT IS 1146
THE DECIMAL LENGTH OF THE INPUT TEXT IS 0240

Use 4 6 8 to calculate the size of the DFF program appended storage. For example, the DFF PAS size for DUMMY2:

= 124 + (2 x 37) + 18 (rounded to next highest 256)
  + 396 (rounded to next highest 256)
= 216 (rounded to next highest 256) + 512
= 256 + 512 = 768

Refer to the following printout of a $CCPDF run:

```
                                $Z0009 DISPLAY FORMAT INFORMATION


   EXECUTION TIME DATA - OUTPUT AREA FORMAT - END POSITIONS FOR RPG PROGRAMS
   -------------------------------------------------------------------------

*  FIELD     FIELD      END     *  FIELD     FIELD      END     *  FIELD     FIELD      END     *
*  NAME      LENGTH   POSITION  *  NAME      LENGTH   POSITION  *  NAME      LENGTH   POSITION  *
*  ------    ------   --------  *  ------    ------   --------  *  ------    ------   --------  *
*  OPCODE     004       0004    *  LENGTH     004       0008    *  TMNAME     006       0014    *
*  $Z0009     006       0020    *

   INPUT AREA FORMAT - END POSITIONS FOR RPG PROGRAMS
   --------------------------------------------------

*  FIELD     FIELD      END     *  FIELD     FIELD      END     *  FIELD     FIELD      END     *
*  NAME      LENGTH   POSITION  *  NAME      LENGTH   POSITION  *  NAME      LENGTH   POSITION  *
*  ------    ------   --------  *  ------    ------   --------  *  ------    ------   --------  *
*  RTCODE     004       0004    *  LENGTH     004       0008    *  TMNAME     006       0014    *
*  - AID-     001       0015    *  NAME@      020       0035    *  NAME#      020       0055    *
*  FIRST#     038       0093    *  SECON@     038       0131    *  THIRD#     038       0169    *
*  FORTH@     038       0207    *  FIFTH#     038       0245    *  SIXTH@     038       0283    *
*  ONE        044       0327    *  TWO        044       0371    *  THREE      044       0415    *
*  FOUR       044       0459    *  FIVE       044       0503    *  SIX        044       0547    *

LENGTH OF OUTPUT RECORD AREA REQUIRED IN DFF PROGRAM
       RPG *SUBR92* - 0020       OTHER - 0012

LENGTH OF INPUT RECORD AREA REQUIRED IN DFF PROGRAM
       RPG *SUBR92* - 0547       OTHER - 0539


   INFORMATION FOR USE DURING CCP ASSIGNMENT STAGE
   -----------------------------------------------

THE DECIMAL LENGTH OF THE FIELD DESCRIPTOR TABLE IS 0396
THE DECIMAL LENGTH OF THE OUTPUT TEXT IS 0463
THE DECIMAL LENGTH OF THE INPUT TEXT IS 0579
```

10 Assume only two tasks (user programs) can run concurrently and the largest programs (DUMMY1 and DUMMY2) are multiple request programs (cannot have multiple copies in main storage concurrently), the worst case for two concurrent user programs is calculated for the following:

1. CCPIVP main storage size is 4096 bytes (no DFF).

2. DUMMY1 main storage size is 8448 (8193 rounded up to the next multiple of 256) (compiler output, including linkage edit) + 1792 (for DFF program control information) or 10,240.

3. DUMMY2 main storage size is 6144 (6140 rounded up to the next multiple of 256) (compiler output, including link-edit) + 768 (for DFF program control information) or 8192 (6912 rounded up to the next multiple of 2048).

4. Using the worst case of DUMMY1 and DUMMY2 in core concurrently, the value of the MINUPA parameter would be 10,240 + 8192 = 18,432 (18K).

Note: If the operating environment is such that DUMMY1 and DUMMY2 would never run concurrently, the value of parameter MINUPA could be reduced (DUMMY1 and CCPIVP run concurrently, DUMMY2 and CCPIVP run concurrently).

11 The size of the TP line buffer for a BSCA line is determined from the largest of the BLKL parameters on the TERMATTR statement. For this line, the line buffer length could be the maximum of 1280 (1146 length of output from $ZOREN rounded up to the next multiple of 256) for maximum performance, or a minimum of 512 to save main storage.

The size of the TP line buffer for an MLTA line is specified in the MAXRECL parameter specified on the MLTALINE statement.

Note: When a line buffer size less than the output text length is encountered by DFF, DFF sends the text in segments equal to or less than the line buffer size. For output text lengths greater than 256, 512 is the smallest line buffer size supported by DFF.

12 The largest value of the BLKL parameter (in the TERMATTR statement) or the MAXRECL parameter (in the MLTALINE statement) per system is used by $CCPAS. $CCPAS calculates the main storage requirements for reserving a minimum PUT buffer in the MINTPBUF area. The reserved PUT area is used only for PUT data operations. If only DFF terminals are defined in the system, the minimum reserved PUT data area is 512 bytes. For a layout of the MINTPBUF, see step **13**.

13 The MINTPBUF parameter specifies the minimum size of the dynamic TP buffer area. This size, rounded up to the next 2K boundary, is the TP buffer area. To determine a value for MINTPBUF to get maximum performance without wasting main storage, the following considerations apply:

TP Buffer Layout

(1)	Reserved PUT data area (See **12**)
(2)	TP PUT or GET (invite) area
(3)	TP Invite parameter list area

At CCP Startup, the TP buffer area is automatically allocated into three areas according to the following algorithm:

Size of (1) is either A or B:

A is 516 (512 data block + 4 getmain segment list) if all terminal attributes are DFF.

B is the greater of MAXRECL + 4 (MLTALINE statement for MLTA terminals) or BLKL + 23 (TERMATTR statement for BSCA terminals) if all terminal attributes in the assignment set are non-DFF. If DFF and non-DFF terminal attributes are mixed, the size of (1) is the greater of A or B.

Size of (3) is (N x 23) + 4, where N equals the number of input capable terminals.

Size of (2) is remainder of area.

CCP requires this area to be large enough to handle the largest system invite, PRUF INVITE, or program request from the system operator console. The area required for program requests from the system operator's console is calculated as follows: = (104 + PCT). (The PCT is the program control table.)

The largest system invite to a terminal is calculated as the maximum of one of the following:

1. PGMREQL + PCT + 4
2. Maximum PRUFLNG + PCT + 7 + 4 (for DFF)
3. Maximum PRUFLNG + PCT + 15 + 4 (for non-DFF)
4. COMMANDL + 4

Note: Add eight additional characters for a 3270 command terminal for items 1 and 4.

The maximum PCT = 35[1] + 4 x (number of disk files used by the program) + 2 x (number of required terminals).

The length of a system invite issued to a terminal in command mode for our example assignment set is calculated as follows:

Maximum PCT = 35[1] + (4 x 2) + (2 x 0) = 42
PGMREQL + PCT = 42 + 15 = 57
COMMANDL = 50
Length of largest system invite = 57 + 4 = 61
If 3270 command terminal 61 + 8 = 69

For the example assignment set with no programs running, the MINTPBUF parameter must be large enough to handle four terminals in command mode. Therefore, the sum of areas (1), (2), and (3) are:

= 516 + (4 x 69) + ((4 x 23) + 4)
(1) (2) (3)
= 516 + 276 + 96 = 888 bytes

However, with this value for MINTPBUF, DUMMY1 and DUMMY2 cannot run because they use $ZOREN and $Z0009 which require 240 and 579 bytes respectively for invite buffers, plus 4 bytes for the GETMAIN parameter list. Therefore, the minimum value of TP buffer size selected should equal 516 + (579 + 4) + 96 = 1195 bytes.

To get maximum performance when DUMMY1 is communicating to all four terminals and the console is being used for CCP commands only (no program requests), the value of MINTPBUF equals:

(1) (2) (3)
516 + (4 x (579 + 4)) + ((4 x 23) + 4)
= 516 + 2332 + 96
= 2944 bytes

Notes:

1. The previous examples do not consider the situation where the system operator enters program requests from the system operator's console. If the system operator is allowed to enter program requests, the size of area (2) must be increased by 104 + PCT size:

   2944 + 104 + 42 = 3090 **13**

2. If two or more lines are being used, the consideration for the worst case in the same environment makes possible additional terminals on the second and succeeding lines. In this case, a larger MINTPBUF parameter value would be required for maximum performance.
3. In the previous example, if the value of the MINTPBUF parameter is less than 3090, the system could run without performance degradation if the largest input format (579) is not used at all four terminals concurrently.
4. If main storage is critical, varying the size of the TP buffer at startup adjusts the size of area (2) to an acceptable size for speed and storage. A momentary lockout might occur until an input or output operation is completed allowing a portion of the teleprocessing buffer to become available.
5. The end main storage address of the TP buffer area is always rounded up to a 2K value; therefore the size of the TP buffer area allocated at startup is always equal to or larger than the MINTPBUF parameter value.
6. If MINTPBUF is not available to handle a DFF PUT operation, automatic blocking is attempted. Under automatic blocking, the next smallest multiple of 256 (minimum of 512) is used if available from TP buffer. In this case, a corresponding TP line buffer is used for this PUT operation. Therefore, even if a TP line buffer of 2048 is defined, if TP buffer at this time has only 768 available, this DFF PUT operation is performed using a block length of 768.

USER SECURITY DATA PROGRAM ($CCPAU)

The intention of the user security data program ($CCPAU) is to take input from the system input device, process it, and, if correct, write it out to the object module $CC4Z9 as the user's security data. This program ($CCPAU) is provided only if SECURE-USER was specified at generation in the $ESEC statement. The object module $CC4Z9 is copied to the CCP production pack only if SECURE-USER was specified at generation in the $ESEC statement. If $CCPAU is loaded and the module $CC4Z9 is not on the production pack, the system issues a VF 4SY halt.

[1] For Program Number 5704-SC1 the value is 35; for Program Number 5704-SC2 the value is 41.

Input is in the form of security data records that can be one of seven types. The type of record is specified by the first position of the input. The type indicator can be any of the following: C, Z, X, I, J, K, and L. In addition, a command input record is allowed and is designated by an asterisk (*) in position 1. Comment records are logged to the system log device.

Each type of input record is intended to define a unique type of data. The following apply to all input data types:

- Positions 2 and 3 define the number of data bytes (including the sign) to be used from the input record and the data must begin in position 5.
- Position 4 is expected to be blank. Some record types can define signed (+ or -) values. In this case the sign, if present, is entered in position 5 and the data begins in position 6. The position occupied by the sign in the input record is counted in the number of input data characters that is specified in positions 2 and 3.

Each input record defines data to be placed in successive locations in the object module $CC4Z9.

The following input data types are accepted by $CCPAU. The data type is signified by the character specified in postion 1:

C Character String
The data can include any character from the extended character set including blanks. The output is the characters specified, in EBCDIC. The output length is the number of bytes specified in positions 2 and 3.

Z Zoned Decimal
The data can include any decimal digits and can be preceeded by + or -. The output is a signed zoned-decimal number. The output length is the number of digits entered (that is, the number of bytes specified in positions 2 and 3, less one if a sign was specified).

X Hexadecimal Data
The count in positions 2 and 3 must be even. Every character of input data must be in the range 0-9, A-F. The output is a set of bit configurations corresponding to the hexadecimal input. The output length is half the number of bytes as were specified in positions 2 and 3.

I One-Byte Binary Integer
The input data is limited to three positions plus an optional sign. The numeric value must not exceed 255. The resultant data will occupy one byte in $CC4Z9, and be represented as a binary integer.

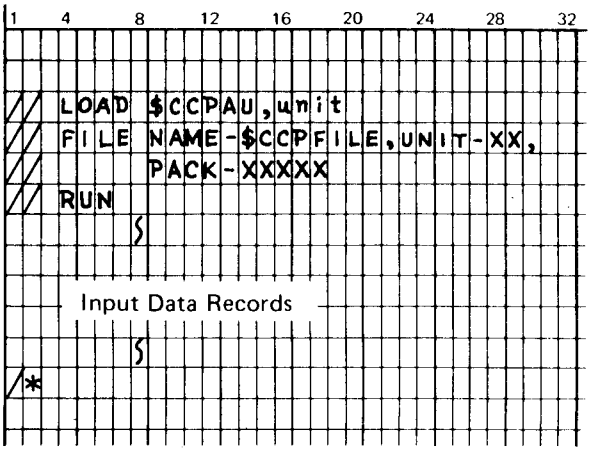
J Two-Byte Binary Integer
The input data is limited to five positions plus an optional sign. The numeric value must not exceed 65535. The resultant output will occupy two bytes in $CC4Z9, and be represented as a binary integer.

K Three-Byte Binary Integer
The input data is limited to eight positions plus an optional sign. The numeric value must not exceed 16777215. The resultant output will occupy three bytes in $CC4Z9, and be represented as a binary integer.

L Four-Byte Binary Integer
The input data is limited to ten positions plus an optional sign. The numeric value must not exceed 4294967295. The resultant output will occupy four bytes in $CC4Z9, and be represented as a binary integer.

$CCPAU continues to accept input data from the system input device until a /* is read, or until enough data to fill $CC4Z9 is accumulated. At this time the data to be written to $CC4Z9 is printed, in hex, on the system log device so that the data can be verified. After printing has finished, a CPU-A3 halt occurs. Option 0 as a reply causes the data to be written to $CC4Z9. Option 2 causes the job to be terminated and $CC4Z9 not to be updated.

The following OCL statements are needed:



For a description of writing a user security program, see *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579.

Chapter 8. System History Area Copy Program (Program Number 5704-SC2 Only)

The System History Area Copy Program, $HACCP, is used to copy the current portion of the System History Area (SHA) to a user-defined disk file. This program only runs under control of CCP.

$HACCP is designed to be automatically invoked when the SHA is nearly full, but it can also be manually invoked by the system operator at any time (using the PF9 key). $HACCP cannot be invoked from a terminal.

Note: The $HIST program can also be used to save the contents of the SHA. However, $HIST requires a batch partition and it cannot be automatically invoked.

USER REQUIREMENTS

On a main data area the user must define a file, named $SHAFILE, with a record length of 128. The format of the records written to $SHAFILE is the same as the format of the records that $HIST writes to the $HISTORY file.

$SHAFILE must also be defined in the CCP assignment set as having consecutive add (CA) data management. This allows subsequent executions of the program to add new records to those already existing in the file. To access the information in $SHAFILE, a user-written program is required.

To use $HACCP, the following statements must be in your CCP assignment set:

```
// DISKFILE NAME-$SHAFILE,ORG-C,RECL-128
// PROGRAM NAME-$HACCP,PACK-PROGRAM,
   PGMDATA-NO,FILES-'$SHAFILE/CA'
```

If $HACCP is to be automatically invoked when the history area is nearly full, then the following statement must also be in the assignment set:

```
// SYSTEM MAXCHAIN-nn
```

$HACCP requires 8K of the CCP user program area.

OPERATING CONSIDERATIONS

The SHA halt status can be changed by using the HALT SHA or NOHALT SHA commands. If $HACCP is to be automatically invoked, the following OCC must be entered:

```
HALT SHA,CCP[,tracks] [1]
```

This condition remains in effect until IPL is performed or until a HALT SHA or NOHALT SHA command is entered.

To permanently establish automatic invocation, the Configuration Record Program ($CNFIG) can be used to set the automatic function with the following statement:

```
// SHA HALT-CCPAUTO[,TRACKS-xx] [1]
```

For further information on how to establish automatic execution, see $CNFIG in *IBM System/3 Model 15 System Control Programming Concepts and Reference Manual*, GC21-5162.

After each execution of $HACCP, an informational message will be issued giving the number of records added to $SHAFILE and the number of records left in $SHAFILE before it fills. If $SHAFILE should fill while the program is copying to it, a decision message will be issued and the file must be emptied before $HACCP can be successfully run again.

If CCP terminates abnormally, the $SHAFILE will not be closed; therefore, records could be lost. See *File Recovery Procedures* in *IBM System/3 CCP System Design Guide*, GC21-5165 for recovery procedures.

[1] The tracks parameter can be any integer value up to 10, but not greater than half the size of the SGA. This parameter is used to set the wraparound warning point of the SHA, and it is not required unless a warning point of 0 is currently in effect.

Appendix A. Generation Control Statement Summary Chart

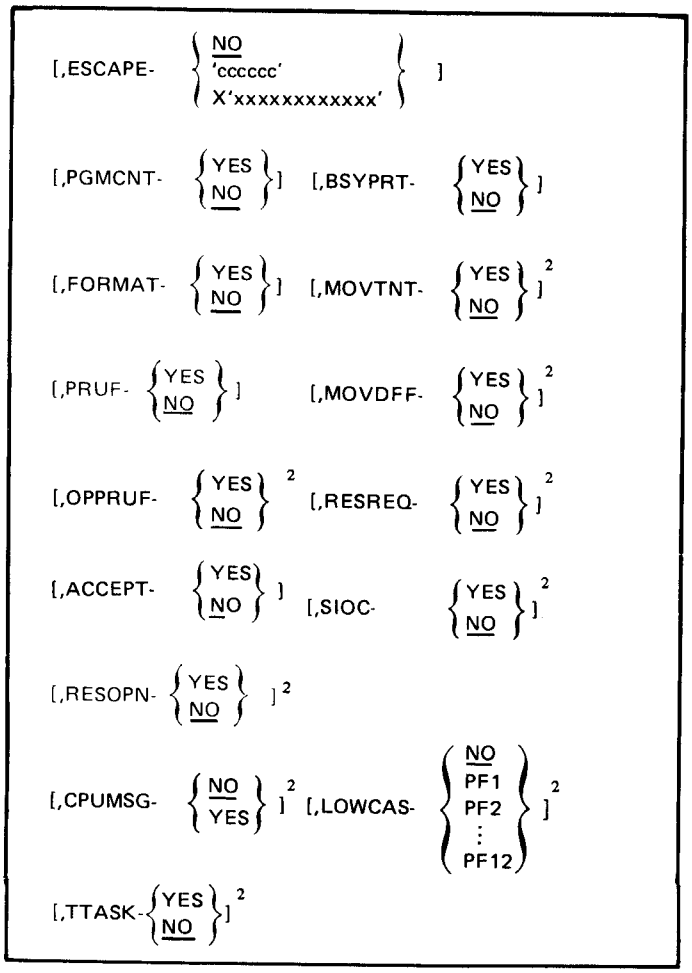
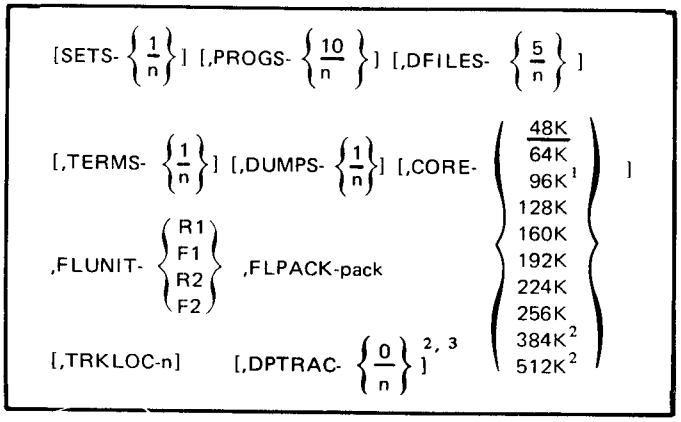
The following is a summary chart containing all valid CCP generation control statements. The statements are listed in alphabetic order. Three items are given for each statement:

- Name
- Format of the statement with all valid operands
- Synopsis of the functional description

For more detailed information on any of the CCP generation control statements, see index entry: *generation control statements.*

Name	Generation Control Statement Description	Function
$EBSC	[BSCA-{0/1/2}] [,DA-{YES/NO}] [,DIAL-{YES/NO}] [,PP-{YES/NO}] [,MP-{YES/NO}] [,CS-{YES/NO}] [,GETMSG-{YES/NO}] [,ITB-{YES/NO}] [,RECSEP-{1E/XX}] [,ASCII-{YES/NO}] [,EBCDIC-{YES/NO}] [RESPOL-{YES/NO}] [,AUTORS-{YES/NO}] [,XPRNCY-{YES/NO}], [INTPOL-{YES/NO}] [,PORT-{YES/NO}][2]	Indicates general specifications concerning binary synchronous communications (BSCA) support.
$EBSD	TYPE-{3275M1, 3277M1, 3284M1, 3286M1, 3275M2, 3277M2, 3284M2, 3286M2, 3735, 3741, CPU}	Indicates a BSC device type that the CCP supports (for BSCA).
$ECSC[2]	[,BSCC-{0/1/2}] [GETMSG-{YES/NO}] [,ITB-{YES/NO}] [,RECSEP-{1E/xx}] [,ASCII-{YES/NO}] [,EBCDIC-{YES/NO}] [,XPRNCY-{YES/NO}] [,INTPOL-{YES/NO}] [,CS-{YES/NO}][2] [,PP-{YES/NO}] [,PORT-{YES/NO}][2]	Indicates general specifications concerning BSCC support.
$ECSD[2]	TYPE-{3275M1, 3277M1, 3284M1, 3286M1, 3275M2, 3277M2, 3284M2, 3286M2, 3735, 3741, CPU}	Indicates a BSC device type that CCP supports. (For BSCC.)

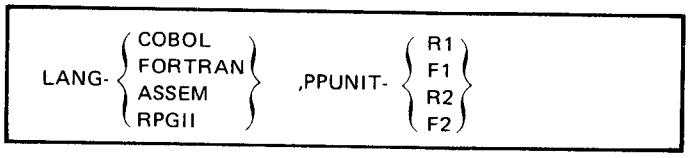
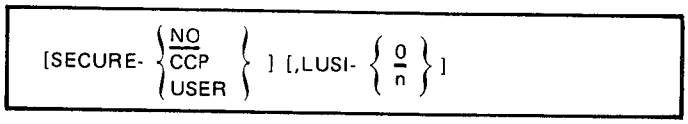
[1] Used with 5704-SC1 only.
[2] Used with 5704-SC2 only.

Name	Generation Control Statement Description	Function
$EFAC		Indicates options that determine the CCP facilities included during generation.
$EFIL		Indicates user options on sizes of items that affect the allocated size and location of $CCPFILE.

[1] Default for Program Number 5704-SC2.
[2] Program Number 5704-SC2 only.
[3] Default is 0. The non-zero minimum value for n is 6. The maximum value is 1000.

Name	Generation Control Statement Description	Function
$EGEN	DSUNIT- {F1 / R1} , [,DPUNIT-unit] [1] CCUNIT- {F1 / R1 / F2 / R2} , [,DPPACK-pack] [1] WKUNIT- {unit / 'unit,unit,unit'} WKPACK- {pack / 'pack,pack,pack'} ,DIUNIT- {F1 / R1 / F2 / R2} [MINRES- {YES / NO}] [,CARD- {NO / YES}]	Indicates where various unit and pack names are located during the CCP generation.
$EMLA	LINES-n[,XLATE- {YES / NO}]	Indicates whether the multiple line terminal adapter (MLTA) is used by the CCP and, if so, the number of MLTA lines. Also indicates whether or not line translation is used on all MLTA operations.
$EMLD	TYPE- {1050 / 1050D / 2740 / 2740S / 2740C / 2740SC / 2740D / 2740DT / 2740DC / 2740DTC / 2740M2S / 2740M2SB / 2740M2SC / 2740M2SCB / 2741 / 2741D / CMCSTD / SYS7C / SYS7SC / SYS7DC} ,XMCODE- {CORR / PTTCEBCD / PTTCBCD}	Indicates an MLTA terminal type supported with its features and the transmission code required on the line.

[1] Program Number 5704-SC2 only.

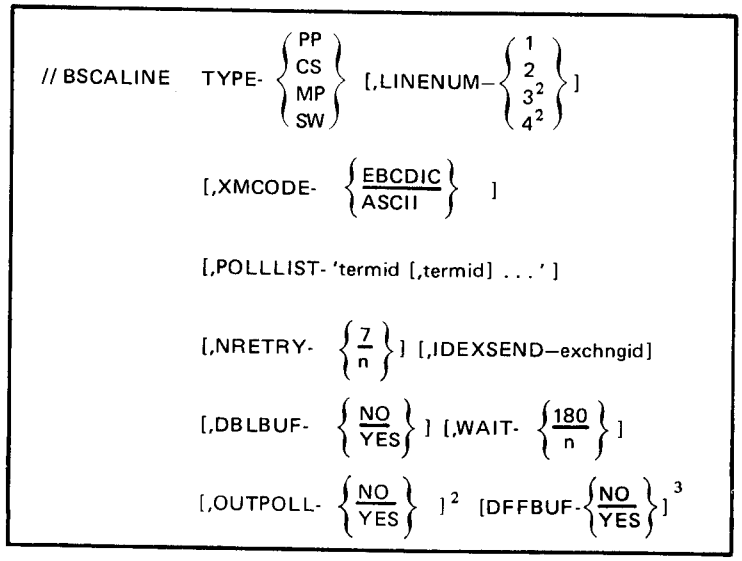
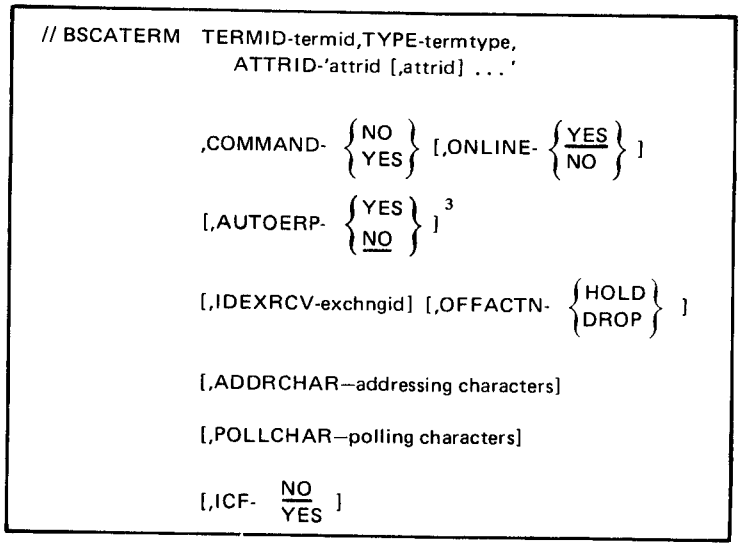
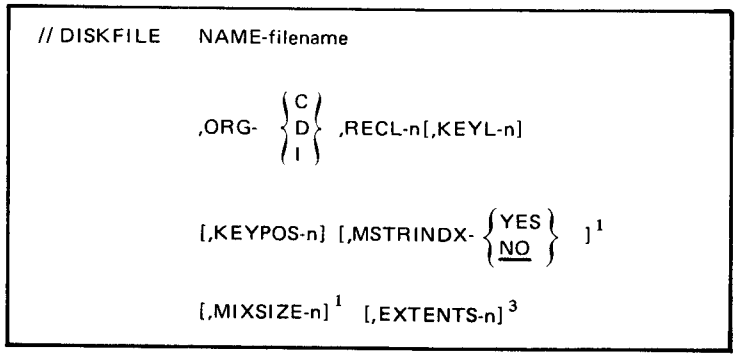
$EPLG		Indicates a programming language the user wishes to use for program preparation under the CCP.
$ESEC		Indicates the type of terminal sign-on security to be used (if any).

Appendix B. Assignment Control Statement Summary Chart

The following is a summary chart containing all valid CCP assignment control statements. The statements are listed in alphabetic order. Three items are given for each statement:

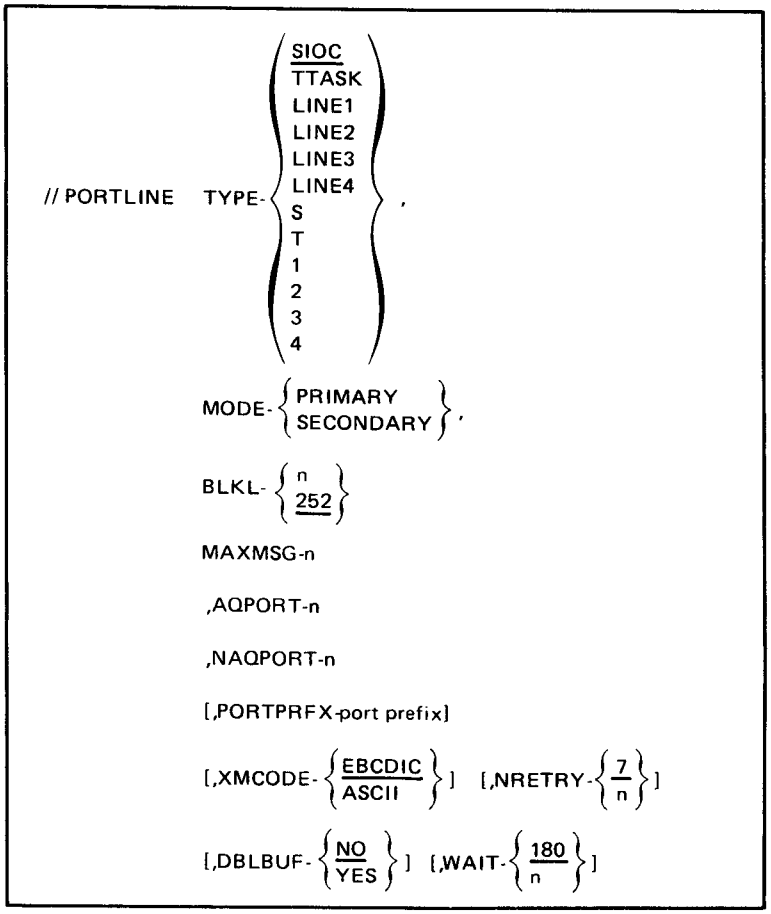
- Name
- Format of the statement with all valid parameters
- Synopsis of the functional description

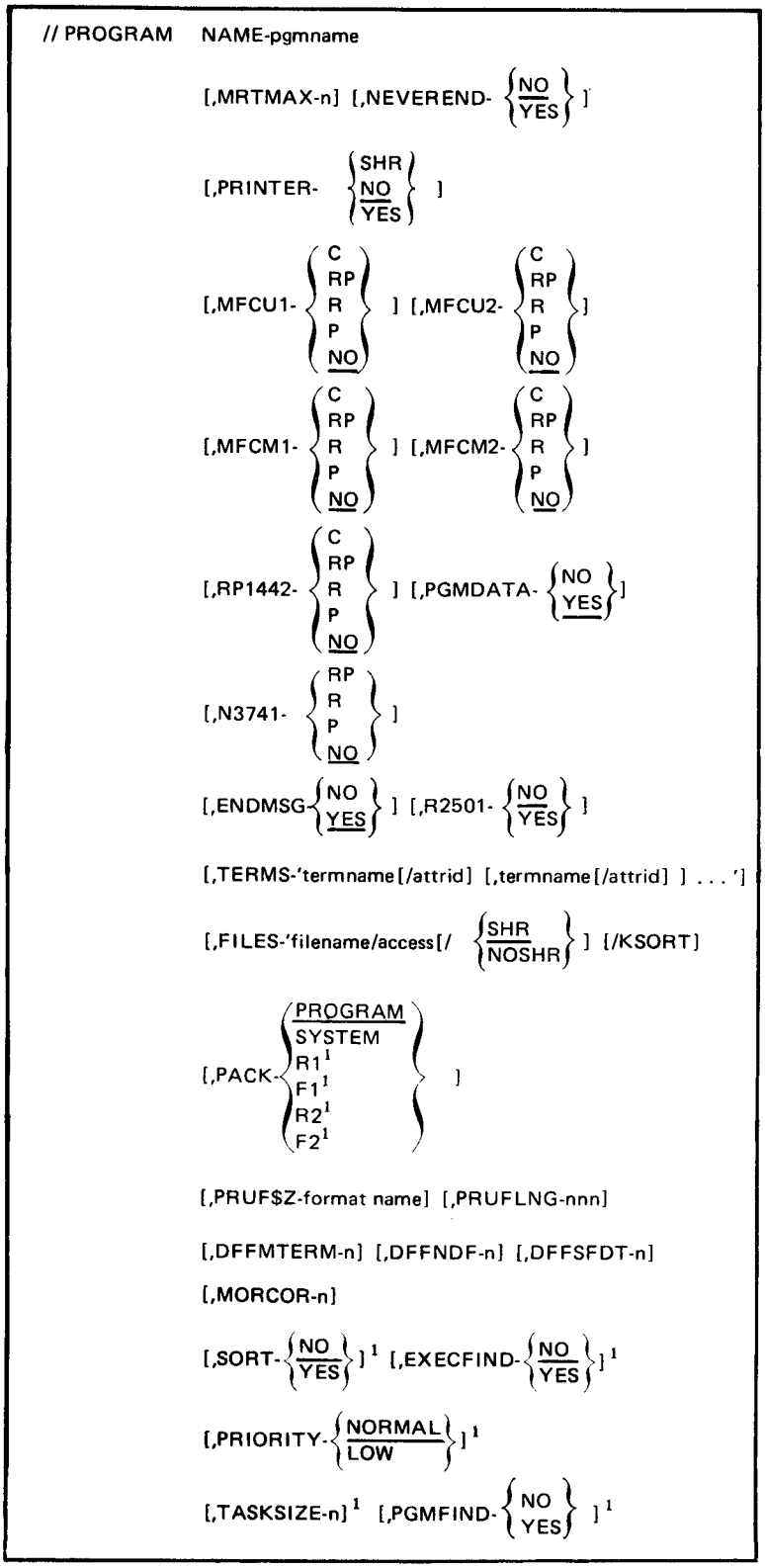
For more detailed information on any of the CCP assignment control statements, see index entry: *assignment control statements*.

Name	Assignment Control Statement Description	Function
BSCALINE		Defines the type of BSC line to be used and the features of the line.
BSCATERM		Defines certain attributes of the terminals on BSCA lines.
DISKFILE		Describes the disk file that will be used during the execution of this set.

[1] Not applicable for Program Number 5704-SC2.
[2] Program Number 5704-SC2 only, TYPE-CS is required.
[3] Program Number 5704-SC2 only.

Name	Assignment Control Statement Description	Function
LIST	// LIST [SET- {id / ALL / DIR / CONFIG}] [PGMSTAT- {NO / YES}] [RESETPS- {NO / YES}]	This is a control statement for $CCPAL that defines which options the user wishes to exercise in listing the contents of the file, $CCPFILE.
MLTALINE	// MLTALINE TYPE- {PP / CS / SW / CW} ,LINENUM- {1 / 2 / 3 / 4 / 5 / 6 / 7 / 8} ,XMCODE- {PTTCEBCD / PTTCBCD / CORR} ,MAXRECL-n[,POLLLIST-'termid[,termid] . . . '] [,DATARATE- {134 / 600 / 1200}] [,AUTOPOLL- {NO / YES}] [,RCVINT- {NO / YES}] [,NRETRY- {2 / n}] [,DELAY-n] [,TIOLT- {YES / NO}]	Defines the type of communication line to be used and the features of the line.
MLTATERM	// MLTATERM TERMID-termid,TYPE-termtype, ,ATTRID-'attrid [,attrid] . . .',COMMAND- {NO / YES} [,ADDR-xx] [,ONLINE- {YES / NO}] [,PINCOMP-n] [,POUTCOMP-n] [,OFFACTN- {HOLD / DROP}]	Defines certain attributes of the terminals on MLTA lines.

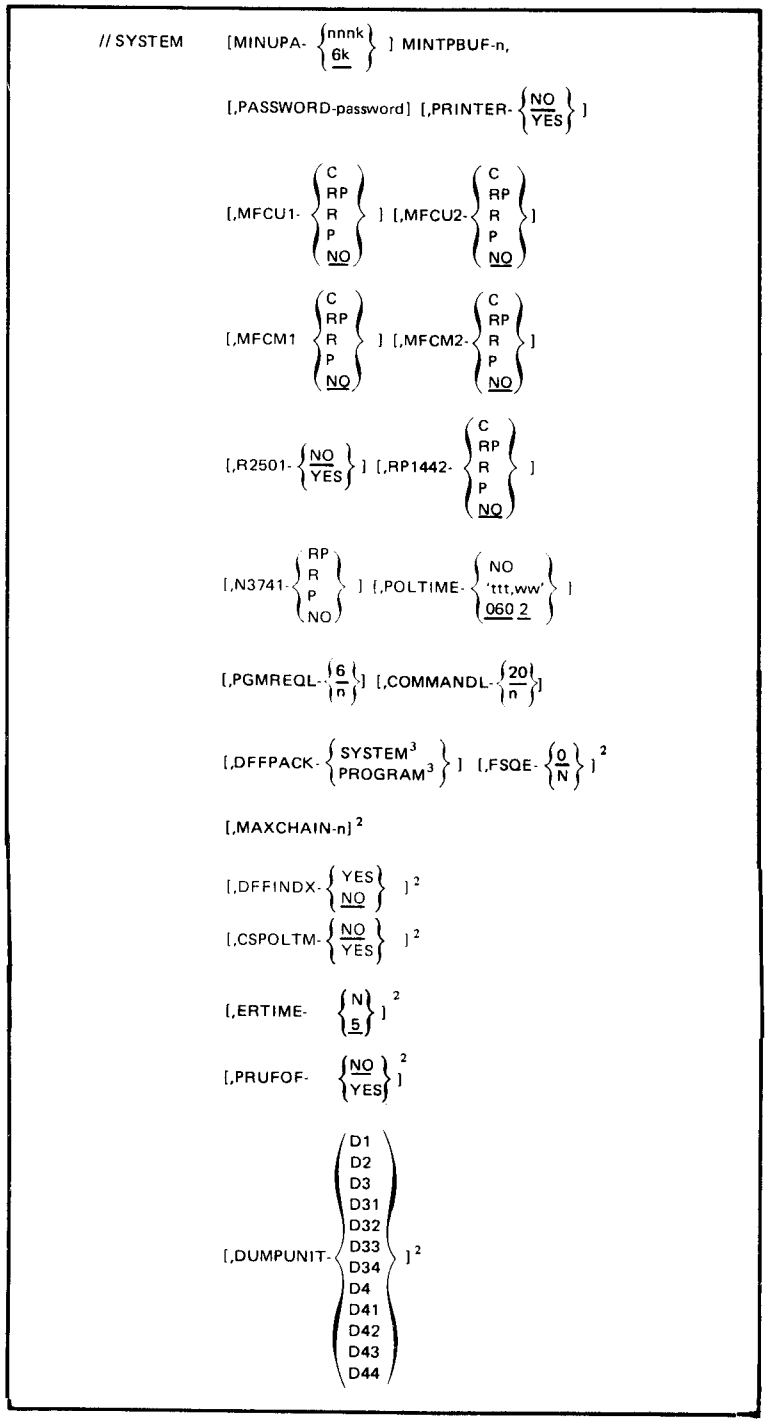
Name	Assignment Control Statement Description	Function
PORTLINE		Defines the type of PORTLINE to be used and the features of the port.

Name	Assignment Control Statement Description	Function
PROGRAM		Defines the logical structure and resource requirements of a user program.

[1] For Program Number 5704-SC2 only.

Name	Assignment Control Statement Description	Function
SET	// SET [ID-{1 / c}] [,ACTION-{CREATE / REPLACE / DELETE / SYSMOD}] [,DFLTEXEC-{YES / NO}] [,ANYSPECS-{YES / NO}][1]	Specifies which set is to be processed and what operation is to occur.
SOURCE	// SOURCE NAME-name,UNIT-{R1 / F1 / R2 / F2} [,END-{YES / NO}]	Specifies that control statements supplied to $CCPAS are read from a source library member.
SYMFILE	// SYMFILE NAME-ccccc,DISKFILE-'ccc . . . [,ccc . . .] . . . '	Defines a symbolic file name and specifies the disk files with which it can be validly associated.

[1] For Program Number 5704-SC2 only.

Name	Assignment Control Statement Description	Function
SYSTEM		Defines parts of the environment in which the CCP will execute.

[1] For Program Number 5704-SC1 only.
[2] For Program Number 5704-SC2 only.
[3] For Program Number 5704-SC2, the following codes may also be specified:
R1, F1, R2, F2
D1A, D1B, D1C, D1D
D2A, D2B, D2C, D2D
D3A, D3B, D3C, D3D
D3E, D3F, D3G, D3H
D4A, D4B, D4C, D4D
D4E, D4F, D4G, D4H

Name	Assignment Control Statement Description	Function
TERMATTR	// TERMATTR ATTRID-attrid The following parameters may be specified for both the MLTA and BSCA terminals [,TRANSLAT-{YES / NO}] [,UPCASE-{YES / NO}] [,SWITCHED-{NO / AC / MC / AA / MA}] The following parameters may be specified for BSCA terminals only [,BLKL-n] [,RECL-n] [,DATAFORM-{RECORD / BLOCK / MESSAGE}] [,TRANSP-{NO / YES}] [,ITB-{NO / YES}] [,VARL-{NO / YES}] [,SPAN-{NO / YES}] [,VERIFYID-{NO / YES}] [,DFF3270-{YES / NO}]	Defines certain attributes of a terminal.
TERMNAME	// TERMNAME NAME-termname[,TERMID-termid] [,MSTRNAME-termname] [,INCOMP-n] [,OUTCOMP-n] [,PHONENUM-number]	Defines symbolic names associated with terminals and subterminals.

Appendix C. Messages

All CCP messages are contained in the *IBM System/3 Communications Control Program Messages Manual*, GC21-5170.

TERMINALS AND FEATURES SUPPORTED

The following terminals can be used with the CCP.

Through the multiple line terminal adapter:

- 1050 Data Communication System
  - Multipoint switched
  - Multipoint nonswitched
- 2740 Communication Terminal Model 1
  - Basic
  - Checking
  - Dial
  - Dial with checking
  - Dial with transmit control
  - Dial with transmit control and checking
  - Station control
  - Station control with checking
- 2740 Communication Terminal Model 2
  - Station control
  - Station control with checking
  - Station control with buffer receive
  - Station control with buffer receive and checking
- 2741 Communication Terminal
  - Basic
  - Switched
- 3767 Communication Terminal (supported as a 2740)
  - Checking (2740 Model 1)
  - Dial with checking (2740 Model 1)
  - Station control with checking (2740 Models 1 and 2)
- 3767 Communication Terminal (supported as a 2741)
  - Basic
  - Switched
- System 7 (supported as a 2740 Model 1)
  - Checking
  - Dial with checking
  - Station control with checking
- Communicating Magnetic Card SELECTRIC® Typewriter (supported as a 2741)
  - Point-to-point switched
- 5100 Portable Computer (supported as a 2741)
  - Basic
  - Switched

With the binary synchronous communications adapter:

- 3270 Information Display System
  - Multipoint (System/3 is control station)
  - See chart for configuration.
- 3735 Programmable Terminal
  - Switched
  - Multipoint (System/3 is control station)
- 3741 Data Station Models 2 and 4
  - Point-to-point switched
  - Point-to-point nonswitched
  - Multipoint (System/3 is control station)
- 5230 Data Collection System (supported as a 3741 Model 2 or 4)
  - Point-to-point switched
  - Point-to-point nonswitched
  - Multipoint tributary
- System/3
  - Point-to-point switched
  - Point-to-point nonswitched
  - Multipoint control station
  - Multipoint tributary
- System/7
  - Point-to-point switched
  - Point-to-point nonswitched
  - Multipoint (System/3 is the control station)
- System/360, System/370
  - Point-to-point switched
  - Point-to-point nonswitched
  - Multipoint (System/3 is the tributary)

With binary synchronous communications controller:

- 3270 Information Display System
  Multipoint (System/3 is control station)
  See chart for configuration.

- 3735 Programmable Terminal
  Multipoint (System/3 is control station)

- 3741 Data Stations Models 2 and 4
  Multipoint (System/3 is control station)

- System/3 (BSCA)
  Multipoint tributary

- System/7
  Multipoint (System/3 is control station)

Terminals that are equivalent to those explicitly supported can also function satisfactorily. The customer is responsible for establishing equivalency. IBM assumes no responsibility for the impact that any changes to the IBM-supplied products or programs might have on such terminals.

3270 Information Display System Multipoint

Device	Model	Character Capacity		Maximum Speed		Attached Via	Max. Cable Distance[2]	Displays/Printers Supported	
		Display	Print Buffer	Print	Transmit[1]			Up to	
3274 Control Unit	51C				9600 bps	BSC		4	3277, 3284, 3286, 3287, 3288
								8	3262, 3278, 3279, 3287, 3289
	1C							32	3262, 3277, 3278, 3284, 3286, 3287, 3288, 3289, 3279
3271 Control Unit	1				7200 bps				3277-1, 3284-1, 3286-1
	2								3277, 3284, 3286, 3288
3275 Display Station	1	480						1	3284-3
	2	1920							
3276 Control Unit Display Station	2	1920						7	3262, 3278-2, 3287, 3289, 3279
3278 Display Station	2	1920				3274, 3276	5000		
3279 Color Display Station	2A								
3277 Display Station	1	480				3271, 3274, Display Adapter	2000		
	2	1920							
3284 Printer	1		480	40 cps			2000		
	2		1920						
3286 Printer	1		480	66 cps			2000		
	2		1920						
3287 Printer	1		1920	80 cps		3271, 3274, 3276, Display Adapter	5000/ 2000		
	2			120 cps					
	1C			80 cps		3274, 3276	5000		
	2C			120 cps					
3288 Line Printer	2		1920	120 lpm[3]		3271, 3274, Display Adapter	2000		
3289 Line Printer	1		1920	120 lpm[3]		3274, 3276	5000		
	2			300 lpm[3]					
3262 Line Printer	3		4096	465 lpm[3]		3274, 3276	5000		
	13			230 lpm[3]					

[1] Nonswitched. 3275 can also transmit at 1200 bps switched.
[2] In feet from control unit.
[3] With 64-character set.

SYSTEM DEVICE AND PROGRAM REQUIREMENTS

Device Requirements

The following is the minimum device configuration necessary for a communications-based system using the CCP:

- 5415 Processing Unit (CPU) with 48K (Program Number 5704-SC1) or 96K (Program Number 5704-SC2) bytes of main storage.
- 5444 Disk Storage Drive Model 2 (Program Number 5704-SC1) or 3340 Direct Access Storage Facility (Program Number 5704-SC2).
- 3277 Display Station Model 1.
- 1403 Printer.
- An input device.
- Multiple Line Terminal Adapter or one Binary Synchronous Communications Adapter (either with minimum data set, line, and terminal configuration).

Notes:
At least 6K (Program Number 5704-SC1) or 8K (Program Number 5704-SC2) main storage must remain available in the CCP partition to perform the operation startup.

A partition size of at least 48K (Program Number 5704-SC2) is required to start up CCP.

During CCP generation and assignment, the main storage available to the program partition in which the function is being performed must be at least 22K (Program Number 5704-SC1) or 48K (5704-SC2).

During CCP generation and assignment, the communications adapter is not required.

Additional Devices Supported

The following device facilities are supported by the CCP:

- 64K[1], 96K, 128K, 160K, 192K, 224K, 256K, 384K[2], or 512K[2] bytes of main storage.
- 5444 Disk Storage Drive — additional Model 2 or Model 3 (Program Number 5704-SC1 only).
- 5445 Disk Storage — 1 to 4 modules for data storage (Program Number 5704-SC1 only).
- 3344 Direct Access Storage (Program Number 5704-SC2 only).
- 3340 Direct Access Storage Facility — 2 to 4 drives.
- 5424 Multi-Function Card Unit.
- 2560 Multi-Function Card Machine.
- 1442 Card Read Punch.
- 2501 Card Reader.
- Multiple Line Terminal Adapter with up to eight lines.
- Binary Synchronous Communications Adapters (BSCA, one or two adapters).
- Local Communications Adapter (LCA).
- Display Adapter (DA).
- Binary Synchronous Communications Controller (BSCC) with Program Number 5704-SC2 only.
- Serial I/O Channel (SIOC) with Program Number 5704-SC2 only (PRPQ Number 5799-WNK).

Note: DA and BSCA-2 are mutually exclusive; LCA and BSCA-1 are mutually exclusive.

CCP operates in any program partition. However, CCP will not operate concurrently in all partitions. Nor will CCP operate concurrently with a program in another program partition that requires *dedicated* use of the processing unit.

System Programs Required

Execution of CCP requires IBM System/3 Model 15 Disk System Management, including all transient modules for the appropriate IOCS.

Generation of CCP requires IBM System/3 Model 15 System Control Programming (5704-SC1 or 5704-SC2), including:

- Programming support for the desired communication adapters: MLMP for the BSCA; Multiple Line Terminal Adapter Feature (PSHRPQ number 5799-WFK).
- Programming support for the interval timer required if using interval polling or the Wait operation code.

No special systems programming requirements exist for the running of system assignments.

For the preparation of application programs, an applicable compiler or assembler is required.

[1] Program Number 5704-SC1 only.
[2] Program Number 5704-SC2 only.

Appendix E. Glossary

This glossary contains only terms that have a special meaning related to the CCP. Other communications and data processing terms used in this publication are defined in *IBM Data Processing Glossary*, GC20-1699.

command interrupt mode: The operating mode of a terminal following data mode escape until the program execution is resumed by a run command (the terminal reenters data mode) or until the terminal is released by a release command (terminal enters command mode).

command mode: The operating mode of a terminal following a successful sign-on, up to and including the program request. Following program termination, a terminal returns to command mode until another program request is made or until sign-off.

command terminal: A terminal that is capable of commanding CCP services related to requesting a program. Terminals are designated command or data terminals at assignment time.

communications management: A major function of the CCP that controls terminal input/output.

communications service subroutine: A relocatable subroutine provided by the CCP that is link-edited to user programs written in RPG II, COBOL, or FORTRAN IV. The subroutine is called by the user program whenever the program requires a communications service, enabling programmers to request communications services in these languages. A separate subroutine is provided for COBOL, FORTRAN IV, and RPG II; a macro is provided for Basic Assembler.

data entry application: A communications-based system application in which terminals are in continuous operation (as opposed to the typical inquiry application). Data entry applications include document preparation (such as invoice writing) and entering data directly into data files from a terminal, such as in creating files.

data mode: The operating mode of a terminal when it is under control of a user program, until the program terminates, the terminal is released by the program, or the data mode escape characters are entered. While in data mode, a terminal is not in direct communication with the CCP.

data mode escape: A special CCP command, consisting of a unique string of six characters entered at a terminal while the terminal is in data mode. The data mode escape command interrupts the execution of the application program and places the terminal in command interrupt mode.

data terminal: A terminal that is not capable of commanding CCP services. A data terminal is always either in standby mode (not polled for input by the CCP) or in data mode (under control of an application program). Also referred to as a noncommand terminal.

disk system management: The group of system programs that control the operation of the IBM System/3 Model 15. Disk System management performs scheduling, input/output control, storage assignment, data management, and related services.

external pointer list: An area in the user program area that is set aside to hold FSBs, XDTs, share DTF address list entries, and incore DFF index entries. The size of the external pointer list (EPL), can be 2K or 4K bytes, depending on the options specified. (See Figure 7, Section 5 in *Appendix F*.) EPL is applicable to Program Number 5704-SC2 only.

file management: A major function of CCP that controls the use of data files by programs running under the CCP.

format find: A program (CCPFMT) that will find a newly created or modified format while running under CCP.

initial mode: The operating mode of a command terminal before a sign-on at the terminal is accepted by the CCP.

inquiry: A communications-based system application in which a request for information is entered from a terminal and a response is returned to the terminal.

inquiry-with-update: A communications-based system application in which records of transactions entered from terminals are used to interrogate and update one or more master files maintained by the system (synonymous with *inquiry and transaction processing*).

multiple requesting terminal (MRT) program: A type of application program under the CCP that can process requests from more than one requesting terminal concurrently.

never-ending program: A user application program that after it is requested, remains in main storage for a relatively long time.

noncommand terminal: See *data terminal*.

order entry application: A form of data entry application in which transactions (such as sales orders) are entered into a data file form remote terminals.

password security option: An optional CCP feature, selected during CCP generation, that requires a terminal operator to enter a predetermined password before the CCP will accept input from the terminal.

physical file: See *symbolic file*.

program management: The major function of the CCP that fetches programs, allocates system resources to programs, purges programs from main storage, and optionally maintains a count of the number of times each application program is requested.

program request: A command, consisting of a program name entered at a terminal or the system operator's console, that causes the CCP to initiate execution of an application program.

program-request count: The optional CCP program management function of maintaining a count of the number of times each application program is requested.

program request under format (PRUF): This function allows DFF to process program request data of any length. If non-DFF, this function allows processing of more than 78 characters (up to the maximum screen size) of program request data.

program-selected terminal: From the point of view of the application program, a terminal that is selected by an application program for input/output, as opposed to a terminal that requested the program (see *requesting terminal*). Program-selected terminals can be either *required* (must be allocated to the program before the program can run) or *acquired* (allocated dynamically to the program as it is running).

requesting terminal: From the point of view of the application program, a terminal that requested the program, as opposed to a terminal that is selected by the program (see *program-selected terminal*). Requesting terminals are always command terminals.

sign-on: The procedure performed at a terminal while it is in initial mode. This procedure may include entering only the sign-on command, or entering the sign-on command with a password or other user-specified security data.

single requesting terminal program: A type of application program under the CCP that can process a request from only one requesting terminal at a time.

symbolic file: A symbolic name in an application program which can, on separate executions of a program, refer to different files, known as *physical files*. A symbolic file is related by the terminal operator to a specific physical file by means of a file command.

system task: A unit of work for the processing unit from the standpoint of the CCP, consisting of a CCP function (as opposed to a user application, or *user task*) that must be performed by the CCP, such as communications management.

task chaining: The process of requesting initiation of a CCP task from within a currently executing CCP task, without requiring system or operator action.

task identification: An identifying character associated with a task that differentiates between that task and other tasks running concurrently under the CCP or DSM.

terminal reference identification: A unique two-character identifier, assigned to each terminal during the CCP assignment stage, that is used by the CCP and the system operator to refer to a specific terminal.

user task: A unit of work for the processing unit, from the standpoint of the CCP, consisting of a user program (as opposed to a system function, or *system task*) that must be executed by the CCP.

Appendix F. Configuration Limitations, Storage Estimates and Performance Considerations

CONFIGURATION LIMITATIONS AND MAIN STORAGE ESTIMATES

The following information can be useful in:

- Estimating the maximum number of programs, files, and terminals that can be supported by a particular CCP configuration.
- Estimating the main storage required for the dynamic TP buffer.
- Estimating CCP startup main storage requirements.
- Estimating the partition size requirement of an executable CCP installation.

Estimating CCP Configuration Limits

The maximum number of programs, files, and terminals that can be supported during a CCP startup depends on the main storage requirements of three parts of CCP. These parts consist of the following:

- Common support code and control tables.
- Communication support code.
- User program area (UPA).

The common support code and control table portion of resident CCP contain two CCP transient areas, $CC4#1, the user security data area, the display format control routine, and various control blocks, all of which depend on generation and assignment options. This part of CCP must be no greater than 16K. Exceeding this limit will cause CCP startup to terminate. Figure 10 or Figure 12 can be used to determine the size of this part of CCP. Should the results obtained from these figures exceed the above limits, the number of terminals, command terminals, files, and/or programs may have to be reduced. An alternative is to eliminate CCP requirements such as program use counts, interval polling, and resident accept input.

The communication support code within resident CCP contains $CCP#2, TRACE support, MLMP support, MLTA support, the line buffers, and the TP buffer. Users of Program Number 5704-SC2 may have additional CCP resident code following the TP buffer. This code may contain the resident pseudo open/close routine, the external pointer list, the resident program request routine, the DFF support code, and the BSCC communication support code. If the display format facility is configured, the last byte of the DFF control routine ($CC4DF) must not exceed a limit of 18K. The TP buffer places two restrictions on the CCP configuration. The TP buffer itself cannot exceed 18K bytes, and the highest address of the TP buffer, always on a 2K boundary, cannot exceed 48K. Figure 9 is a guideline for approximating the size of TP buffer. This value should be initially used for the MINTPBUF key word on the assignment system statement, but may require adjustment after various CCP startups to obtain the best response times for all terminals. Figure 11 or Figure 13 can be used to determine the main storage requirements for CCP communication support code. The total storage requirements of CCP common code, the control tables, as well as the communication support code make up resident CCP.

The user program area immediately follows the CCP resident code and extends to the end of the CCP partition. The user program area contains the user's programs, any appropriate DSM modules included when link-editing the application programs, storage requirements for memory resident overlays and external buffers, and the additional storage requirements of the display format facility. Additional storage is required within the user program area if the display format facility (DFF) is used to support the 3270 terminal system. This storage is called program appended storage (PAS). The size of the CCP partition and/or minimum user program area (MINUPA parameter within the assignment set) may be adjusted to allow the CCP configuration to fit.

Using This Appendix

The figures within this appendix are useful in making storage estimates and checking for any limitations of CCP. These figures are constructed as a list of options with the corresponding main storage (given in number of decimal bytes) needed for that option. The option so far as possible is given as a keyword with a reference to a particular statement for CCP generation or assignment. If any part of a listed option meets the requirement of the desired system configuration, then the main storage on that line in the table should be added to the system size. The options that are indented are to be included in the system size only if the desired system configuration also includes the preceding nonindented option. This appendix is designed to be used top down. Main storage totals obtained in the latter figures depend upon totals obtained in some of the preceding figures. Users of Program Number 5704-SC1 will use Figures 7, 9, 10, 11, 14, and 15. Users of Program Number 5704-SC2 will use Figures 8, 9, 12, 13, 14, and 15. A general explanation of each figure follows:

- Figure 7 (5704-SC1 only) is used to estimate the sizes of two portions of CCP resident code referred to as common support code and communication support code. These two portions are known internally as $CC4#1 and $CC4#2. Both are totally dependent upon CCP generation specifications and options. The common support code ($CC4#1), and the communication support code ($CC4#2) are loaded into storage at separate phases of CCP startup. Totals obtained from this figure will be used in Figures 10 and 11.
- Figure 8 (5704-SC2 only) is used to estimate the sizes of three portions of CCP resident code referred to as common support code, communication support code, BSCC support code. These three portions are known internally as $CC4#1, $CC4#2, and $CC4#M. They are totally dependent upon CCP generation specifications and options. The common support code ($CC4#1), the communication support code ($CC4#2), and the BSCC support code ($CC4#M) are loaded into storage at separate phases of CCP startup. Totals obtained from this figure will be used in Figures 12 and 13.
- Figure 9 is useful in estimating the size of the dynamic TP buffer and verifying that it does not exceed a limit of 18K. The calculated size of the TP buffer is used in Figure 11 (5704-SC1) or Figure 12 (5704-SC2).
- Figure 10 (5704-SC1 only) is used to estimate the size of CCP common support code and control tables. This portion of CCP resident storage contains the transient areas, $CC4#1, user security routine, and control blocks all of which depend on CCP generation and assignment options. The CCP components listed are, as near as possible, in the same sequence as they are positioned in main storage. This part of CCP must be no greater than 18K for a DFF system. Exceeding this limit will cause startup to terminate. The total obtained from this figure is used in Figure 11.
- Figure 11 (5704-SC1 only) is used to estimate the size of resident CCP. The total obtained within this figure includes all of CCP resident code, all control tables, line buffers, the TP buffer, MLMP and MLTA support, and any trace support. This total cannot exceed 48K. The CCP components listed are, as near as possible, in the same sequence as they are positioned in main storage. The total obtained can be used in Figure 15.
- Figure 12 (5704-SC2 only) is used to estimate the size of CCP common support control tables. This figure is functionally the same as Figure 10 and has the same storage limitation. The CCP components listed are, as near as possible, in the same sequence as they are positioned in main storage. The total obtained from this figure is used in Figure 13.
- Figure 13 (5704-SC2 only) is used to estimate the size of resident CCP. The total obtained within this figure includes all of CCP resident code, all control tables, line buffers, the TP buffer, MLMP and MLTA support, trace support, resident open/close, resident program request, and BSCC communication support. The CCP components listed are, as near as possible, in the same sequence as they are positioned in main storage. The total obtained can be used in Figure 15.
- Figure 14 is used to estimate the storage within the user program area required for DFF program appended storage (PAS).
- Figure 15 is used to determine the total executable CCP partition size. This includes all of resident CCP and the user program area.

The specifications made in an assignment set represent the standard operating environment for a CCP run. Certain resources specified in an assignment set as being available to the CCP run can, on an exception basis, be suppressed by the system operator at the startup of the CCP run. Such a suppression reduces the control block main storage requirements as defined in Figure 10 or Figure 12.

Add only the items below that pertain to your CCP generation specifications and options.	$CC4#1	$CC4#2
Common to all generations . . .	4753	2221
Multiple MLTA and/or BSCA lines (see note 1)	–	23
ESCAPE-'YES' ($EFAC)	17	18
ACCEPT-YES ($EFAC)	272	–
CPUMSG-NO ($EFAC)	33	–
MLTA ($EMLA)	124	5347
XLATE-NO ($EMLA)	–	52
ESCAPE-YES ($EFAC)	–	74
TYPE-(dialed feature terminals) ($EMLD)	–	282
TYPE-(station control terminals)	–	823
TYPE-(1050 terminals)	–	4
TYPE-(no checking basic terminals)	–	230
TYPE-(checking terminals)	–	217
TYPE-1050 or 1050D	4	72
TYPE-1050D	–	54
TYPE-1050	–	3
TYPE-1050, 2740SC, 2740M2SC or 2740SCB	–	51
TYPE-2740SC, 2740M2SC, 2740M2SCB or SYS7SC	–	3
TYPE-2740S, 2740M2S or 2740M2SB	–	54
TYPE-2740	–	54
TYPE-2740D	–	54
TYPE-2740C or SYS7C	–	54
TYPE-2740DC or SYS7DC	–	54
TYPE-2740DT	–	54
TYPE-2740DTC	–	54
TYPE-2741, 2741D or CMCSTD	–	40
TYPE-2741	–	54
TYPE-2741D or CMCSTD	–	54
XMCODE-CORR ($EMLD)	9	–
XMCODE-PTTCEBCD	9	–
XMCODE-PTTCBCD	9	–
BSCA ($EBSC)	146	2135
FORMAT-YES ($EFAC)	1744	3400
PRUF-YES	–	35
PRUF-YES	–	84
BSYPRT-YES ($EFAC)	52	103
ESCAPE-'YES'	3	27
TYPE-3270 ($EBSD)	–	6
TYPE-CPU ($EBSD)	–	6
MLTA ($EMLA)	–	109
TYPE-(STATION CONTROL TERMINALS) ($EMLD)	–	4
BSCA-2 ($EBSC)	–	203
MULTIPLE BSCA and/or MLTA LINES	–	97
MLTA ($EMLA)	–	3
CS-YES or DIAL-YES ($EBSC)	–	208
PP-YES, MP-YES or DIAL-YES	–	6
CS-YES	–	166
RESPOL-YES	–	1363
PP-YES, MP-YES or DIAL-YES ($EBSC)	–	6
MP-YES	–	–
PP-YES, CS-YES or DIAL-YES	–	6
AUTHORS-YES	–	329

Figure 7 (Part 1 of 2). Estimating Storage Requirements for CCP Common Support Code ($CC4#1) and Communication Support Code ($CC4#2) – 5704-SC1 only

	$CC4#1	$CC4#2
BSCA ($EBSC) (continued)		
DIAL-NO		
DIAL-YES	–	97
MP-YES, CS-YES or PP-YES	–	547
MP-YES	–	144
MLTA ($EMLA)	–	3
PP-YES or MP-YES ($EBSC)	–	6
DA-YES ($EBSC)	–	6
DIAL-YES	–	418
DIAL-NO	–	4400
3270 ONLY	–	–
MINRES-NO and RESPOL-YES	–	3314
GETMSG-YES ($EBSC)	–	341
ITB-YES ($EBSC)	–	91
XPRNCY-YES ($EBSC)	–	19
XPRNCY-YES	–	24
ASCII-YES ($EBSC)	–	13
TYPE-3270 ($EBSD)	6	92
TYPE-CPU	–	23
INTPOL-YES ($EBSC)	–	23
TYPE-CPU ($EBSD)	–	184
TYPE-3270, 3741 or 3735	–	32
TYPE-3270 or 3741	–	59
TYPE-3741	–	29
ASCII-YES	3	2
TYPE-3270 or 3735	–	2
TYPE-3735	–	3
ASCII-YES ($EBSC)	–	23
TYPE-3270 ($EBSD)	–	14
DIAL-YES ($EBSC)	–	353
CS-YES	–	16
CS-YES ($EBSC)	–	6
DIAL-YES	–	9
MLTA ($EMLA)	–	6
ESCAPE-'YES' ($EFAC)	–	6
ASCII-YES ($EBSC)	–	7
TYPE-3741 ($EBSD)	–	28
ESCAPE-'YES' ($EFAC)	–	3
PRUF-YES ($EFAC)	–	6
MINRES-NO ($EGEN)	–	3
DIAL-YES	–	384
PP-YES, MP-YES or CS-YES	–	3
GETMSG-YES ($EBSC)	–	3
ITB-YES ($EBSC)	–	8
XPRNCY-YES ($EBSC)	–	25
MINRES-YES ($EGEN)	–	31
	6	–
Totals . . . See note—	[2]	[3]

Notes:

1. Include if number of MLTA lines plus number of BSCA lines is greater than one.
2. Also enter total for $CC4#1 in Figure 4, Section 1.
3. Also enter total for $CC4#2 in Figure 5, Section 1.

Figure 7 (Part 2 of 2). Estimating Storage Requirements for CCP Common Support Code ($CC4#1) and Communication Support Code ($CC4#2) – 5704-SC1 only

Add only the items below that pertain to your CCP generation specifications and options.	$CC4#1	$CC4#2	$CC4#M
Common to all generations . . .	4732	575	–
Multiple MLTA and/or BSCA lines (see note 1)	–	23	–
FORMAT-YES ($EFAC)	1770	–	–
MOVTNT-YES	14	–	–
ACCEPT-YES	315	–	–
RESOPN-YES	32	–	–
RESREQ-NO	22	–	–
ESCAPE-YES	20	–	–
MOVTNT-YES	164	–	–
BSCA ($EBSC) or MLTA ($EMLA)	–	1825	–
BSCC ($ECSC)	–	7	–
ESCAPE-YES ($EFAC)	–	18	–
MINRES-NO ($EGEN)	–	4	–
MLTA ($EMLA)	66	5342	–
XLATE-NO ($EMLA)	–	56	–
ESCAPE-'YES' ($EFAC)	–	80	–
TYPE-(dialed feature terminals) ($EMLD)	–	282	–
TYPE-(station control terminals)	–	827	–
TYPE-(1050 terminals)	–	4	–
TYPE-(no checking basic terminals)	–	230	–
TYPE-(checking terminals)	–	217	–
TYPE-(buffered receive terminals)	–	39	–
TYPE-1050 or 1050D	75	72	–
TYPE-1050D	–	54	–
TYPE-1050	–	3	–
MOVTNT-YES	22	–	–
TYPE-1050, 2740SC, 2740M2SC or 2740SCB	–	51	–
TYPE-2740SC, 2740M2SC, 2740M2SCB or SYS7SC	–	3	–
TYPE-2740S, 2740M2S or 2740M2SB	–	54	–
TYPE-2740	–	54	–
TYPE-2740D	–	54	–
TYPE-2740C or SYS7C	–	54	–
TYPE-2740DC or SYS7DC	–	54	–
TYPE-2740DT	–	54	–
TYPE-2740DTC	–	54	–
TYPE-2741, 2741D or CMSTD	–	40	–
TYPE-2741	–	54	–
TYPE-2741 or CMSTD	–	54	–
BSCA ($EBSC)	146	5335	–
MLTA ($EMLA)		98	–
BSCA-2 ($EBSC)	–	203	–
DA-YES ($EBSC)	–	418	–
Multiple BSCA and/or MLTA lines (see note 1)	–	105	–
INTPOL-YES ($EBSC)	–	201	–
XPRNCY-YES	–	31	–
ITB-YES ($EBSC)	–	18	–
XPRNCY-YES	–	24	–
BSYPRT-YES	52	103	–

Figure 8 (Part 1 of 3). Estimating Storage Requirements for CCP Common Support Code ($CC4#1), Communication Support Code ($CC4#2), and BSCC Support Code ($CC4#M) – 5704-SC2 only

BSCA ($EBSC) (continued)	$CC4#1	$CC4#2	$CC4#M
ASCII-YES ($EBSC)	–	94	–
TYPE-3270 ($EBSD)	–	51	–
TYPE-CPU	–	12	–
TYPE-3735	–	14	–
TYPE-3741	3	13	–
MINRES-NO ($EGEN)	–	375	–
DIAL-YES ($EBSC)	–	3	–
PP-YES, MP-YES or CS-YES	–	6	–
GETMSG-YES ($EBSC)	–	8	–
ITB-YES ($EBSC)	–	22	–
XPRNCY-YES	–	26	–
TYPE-3270, 3741 or 3735	–	5	–
MLTA ($EMLA)	–	3	–
PRUF-YES ($EFAC)	–	16	–
RESPOL-YES	–	337	–
FORMAT-YES ($EFAC)	9	3430	–
PRUF-YES	–	35	–
PRUF-YES ($EFAC)	–	84	–
GETMSG-YES ($EBSC)	–	94	–
ESCAPE-'YES' ($EFAC)	–	39	–
TYPE-3270 ($EBSD)	–	6	–
TYPE-CPU ($EBSD)	–	6	–
TYPE-3270, 3741 or 3735 ($EBSD)	–	59	–
TYPE-3270 or 3741	–	29	–
TYPE-3270 or 3735	–	3	–
TYPE-3270 or ESCAPE-'YES'	–	7	–
TYPE-3735 ($EBSD)	–	23	–
TYPE-3741	–	60	–
TYPE-3270	–	350	–
DIAL-YES ($EBSC)	–	16	–
CS-YES ($EBSC)	–	3	–
DIAL-YES	–	18	–
MLTA ($EMLA)	–	9	–
ESCAPE-'YES' or PRUF-YES	–	3	–
ESCAPE-'YES' ($EFAC)	–	5	–
PRUF-YES	–	3	–
DIAL-YES or TYPE-3735, 3741 or CPU	–	742	–
MINRES-YES or RESPOL-NO	–	337	–
TYPE-CPU ($EBSD)	–	32	–
CS-YES or DIAL-YES ($EBSC)	–	208	–
PP-YES, MP-YES or DIAL-YES	–	6	–
DIAL-YES	–	587	–
MP-YES, CS-YES or PP-YES	–	42	–
MP-YES	–	3	–
MLTA ($EMLA)	–	6	–
CS-YES	–	172	–
PP-YES, MP-YES or DIAL-YES ($EBSC)	–	18	–
DIAL-YES ($EBSC)	–	6	–
RESPOL-YES	–	1893	–

Figure 8 (Part 2 of 3). Estimating Storage Requirements for CCP Common Support Code ($CC4#1), Communication Support Code ($CC4#2), and BSCC Support Code ($CC4#M) – 5704-SC2 only

		$CC4#1	$CC4#2	$CC4#M
BSCA ($EBSC) (continued)				
PP-YES, MP-YES or CS-YES ($EBSC)		–	84	–
PP-YES, MP-YES or DIAL-YES ($EBSC)		–	6	–
PP-YES or MP-YES ($EBSC)		–	6	–
MP-YES		–	18	–
PP-YES, CS-YES or DIAL-YES		–	6	–
AUTORS-YES		–	329	–
BSCC ($ECSC)		52	–	11105
FORMAT-YES ($EFAC)		–	–	65
BSCC-2 ($ECSC)		–	–	56
INTPOL-YES ($ECSC)		–	–	43
BSCC-2 ($ECSC)		–	–	10
ESCAPE-'YES' ($EFAC)		–	–	73
ASCII-YES ($ECSC)		–	–	38
GETMSG-YES ($ECSC)		–	–	13
BSYPRT-YES		–	–	111
SIOC		94	–	1162
BSCC-NO		52	–	11105
MOVTNT-YES		4	–	–
Totals . . .	See note —	2	3	4

Notes:

1. Include if number of MLTA lines plus number of BSCA lines is greater than one.
2. Also enter total for $CC4#1 in Figure 12, Section 1.
3. Also enter total for $CC4#2 in Figure 13, Section 1.
4. Also enter total for $CC4#M in Figure 13, Section 10.

Figure 8 (Part 3 of 3). Estimating Storage Requirements for CCP Common Support Code ($CC4#1), Communication Support Code ($CC4#2), and BSCC Support Code ($CC4#M) – 5704-SC2 only

Follow steps one through six to estimate the size of the dynamic TP buffer. Refer to system, TERMATTR, PORTLINE, and program statements within the CCP assignment set.

Put Area

1 A If any TERMATTR statement is DFF (DFF 3270-YES), enter 516

B If any TERMATTR statement is NON-DFF or if Program Number 5704-SC2 and any BSC line has an optional DFF buffer, enter the longer of RECL+4 or BLKL+23. = _____

Common Area

2 Refer to the program statements and enter the maximum result of;
(2 x the number of terminals allocated to any task) + (4 x the number of disk files allocated to the same task) + 34. = _____

3 Enter the maximum result of A, B, or C below.
A DFF request length (PRUFLNG) + 19
B Program request length (PGMREQL) + 12
C Command length (COMMANDL) + 12 = _____

4 For 5704-SC2 with SIOC only
Enter the result of;
(value of MAXMSG specified on PORTLINE statement) + 15 + (the value calculated in step 2 above). = _____

Invite Area

5 Enter the result of A or B below.
A (number of input capable terminals x 23) + 4 (5704-SC1)
B (number of input capable terminals x 20) + 4 (5704-SC2) = _____

Total TPBUFF

6 Add items 1 through 5 above, enter result. The result should not exceed 18K (18,432) and if Program Number 5704-SC2 using optional DFF buffer, the minimum is 2K (2,048). If so, enter result in Section 3 of Figure 11 (5704-SC1) or Section 4 of Figure 13 (5704-SC2).

_____

Figure 9. Estimating Storage Requirements for the Dynamic TP Buffer

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 1—General Applies to all CCP configurations		
Transient areas	T1 & T2	Space reserved for CCP transients. 1024
Common CCP code	$CC4#1	Enter $CC4#1 storage requirements from Figure 7 here. ____
Program request count table	PCT	If PGMCNT-YES ($EFAC), enter 2 x (number of program statements in assignment set). ____
User security	$CC4Z9	If secure-user is used in $ESEC statement, enter size of user's security information (value of LUSI keyword). ____
Terminal attribute set	TAS	Enter 5 x (number of TERMATTR statements in assignment set). ____
Section 1 total		Add results within this section, enter total here and in Section 8 below. ____
Section 2—Control information for MLTA communications lines. If no MLTALINE statements in assignment set, skip to Section 3.		
		Refer to MLTALINE and MLTATERM statements within assignment set.
Define file/line control block	DTF/LCB	Enter 126 bytes for each MLTALINE statement. MLTA line numbers . . . 1 2 3 4 5 6 7 8 ____ ____ ____ ____ ____ ____ ____ ____
Terminal statistics tables	TST	Enter 5 x (number of MLTATERM statements) for each MLTALINE statement. ____ ____ ____ ____ ____ ____ ____ ____
Polling list (note 1)	POLLLIST	Enter 12 x (number of 1050 terminal entries in POLLLIST) for each MLTALINE statement. ____ ____ ____ ____ ____ ____ ____ ____ Also enter 6 + (3 x number of non-1050 terminal entries in POLLLIST) for each MLTALINE statement. ____ ____ ____ ____ ____ ____ ____ ____

Figure 10 (Part 1 of 6). Storage Requirements for Common Code and Control Tables – 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
		MLTA line numbers . . . 1 2 3 4 5 6 7 8
Check list		Enter 3 for each MLTALINE statement. ___ ___ ___ ___ ___ ___ ___ ___
MLTA line buffers (note 1)		Enter the appropriate values as calculated below for each terminal type on each MLTA line.
		2741: MAXRECL + 15 = ___ ___ ___ ___ ___ ___ ___ ___
		1050 or 2740 automatic polling: MAXRECL + 14 = ___ ___ ___ ___ ___ ___ ___ ___
		1050 not automatic polling: MAXRECL + 17 = ___ ___ ___ ___ ___ ___ ___ ___
		2740 control station, not automatic polling: MAXRECL + 17 = ___ ___ ___ ___ ___ ___ ___ ___
		2740 transmit control: MAXRECL + 17 = ___ ___ ___ ___ ___ ___ ___ ___
		All others terminal types: MAXRECL + 15 = ___ ___ ___ ___ ___ ___ ___ ___
Line totals		Add results above for each MLTA line. ___ ___ ___ ___ ___ ___ ___ ___
Adaptor DTF	DTF	Main storage requirement for MLTA adaptor DTF. + 33
Section 2 total		Add line totals and adaptor requirements. Enter result here and in Section 8 below. ___
		Note 1 — The MLTA POLLLIST and line buffers are actually located just before $CC4#2.

Figure 10 (Part 2 of 6). Storage Requirements for Common Code and Control Tables — 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements	BSCA line number. 1	2
Section 3—Control information for BSCA communications lines. If no BSCALINE statements in assignment set, skip to Section 4.				
		Refer to BSCALINE and BSCATERM statements within assignment set.		
Define file/line control block	DTF/LCB	Enter 136 bytes for each BSCALINE statement.	____	____
		If TYPE-SW within BSCALINE statement, enter 4 for those line numbers.	____	____
Addressing list	ADDRLIST	If TYPE-CS in BSCALINE statement, enter ((number of BSCATERM statements) x (number of ADDRCHAR characters + 3) + 1) for each line statement.	____	____
Polling list	POLLLIST	If TYPE-CS in BSCALINE statement, enter ((number of POLLLIST entries) x (number of POLLCHAR characters + 3) + 1) for each line statement.	____	____
Terminal statistics tables	TST	Enter 4 + ((number of BSCATERM statements) x 2 x (number of POLLCHAR + 7)) for each line statement.	____	____
Check list		Enter 3 for each BSCALINE statement.	____	____
Line totals		Add results above for each BSCA line.	____	____
Section 3 total		Add both line totals, enter result here and in Section 8 below.		____

Figure 10 (Part 3 of 6). Storage Requirements for Common Code and Control Tables — 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 4—Control information for BSCA and/or MLTA terminals.		
		Refer to MLTATERM, BSCATERM and TERMNAME statements within assignment set.
Terminal unit block	TUB	Enter 38 x (number of MLTA and BSCA terminals). ____
		Also enter 19 x (number of BSCA switched terminals) + 1 ____
		Also enter 19 x (number of BSCA command terminals) + 1 ____
Terminal name table	TNT	Enter 23 + (11 x number of TERMNAME statements). ____
Section 4 total		Add results within this section, enter result here and in Section 8 below. ____
Section 5—Control information for symbolic and associated symbolic files. If no SYMFILE statements within assignment set, skip to Section 6.		
		Refer to SYMFILE, BSCATERM and program statements within assignment set.
File specification blocks	FSB	Enter (number of SYMFILES) x ((1 + number of command terminals) + (number of user programs allowed to run concurrently*)). * 15 maximum ____
Symbolic file table	XDT	Calculate the size of one XDT for each SYMFILE statement. Each table = 9 + number of DISKFILE references within SYMFILE statement. ____
		Enter total for all SYMFILE statements. The XDT is actually located following the SDF address list below. ____
Section 5 total		Add results within this section, enter total here and in Section 8 below. ____

Figure 10 (Part 4 of 6). Storage Requirements for Common Code and Control Tables – 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 6—Control information for disk files. If no diskfile statements within assignment set, skip to Section 7.		
Short DTF	SDF	Choose the appropriate following lines according to diskfile statement organizations in the assignment set. The control blocks are positioned in main storage in the same sequence as the diskfile statements in the assignment set. Enter 20 X (number of direct diskfile statements). ____ Enter 21 X (number of consecutive diskfile statements). ____ Enter 30 X (number of indexed load diskfile statements) + 2 x (the key length for each indexed file). ____ Enter 38 X (number of indexed random diskfile statements) + 2 x (the key length for each indexed file). ____ If track index exists, enter 3 x (number of indexed add files). ____
Short DTF address list	SDF @ LIST	Enter 11 X (number of physical files). ____
File sharing queue element	FSQE	Enter 15 X the number of FSQE's specified in the assignment set, or Calculate the following ratio for each program. R = (number of shared files)/(number of 2K blocks required by program) Use the largest calculated ratio for 'R' below. Enter 15 X R X (the maximum number of copies of the program used to obtain R that can fit into the user program area. ____
CCP master indexes		Calculate one master index for each diskfile statement with MSTRINDX-YES specified. Master index length = value specified for MIXSIZE keyword within diskfile statement. or Master index length = (KEYL + 2) x [(number of disk cylinders in index) + 3)]. Enter total for all master indexes here. ____
Section 6 total		Add results within this section, enter total here and in Section 8 below. ____

Figure 10 (Part 5 of 6). Storage Requirements for Common Code and Control Tables — 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 7—Control information for user programs.		
Program characteristics table	PCT	Enter 6 X the number of sectors of PCT in $CCPFILE, or as a rough approximation, enter 1.5 X number of program statements rounded up to the nearest multiple of 6. Also enter value in Section 8 below. _____
Section 8—CCP common code and control table totals.		
Common code and control table total		Enter total from Section 1 here. _____ Enter total from Section 2 here. _____ Enter total from Section 3 here. _____ Enter total from Section 4 here. _____ Enter total from Section 5 here. _____ Enter total from Section 6 here. _____ Enter total from Section 7 here. _____ Add all lines above. Sum represents size of all CCP common code and control tables. It cannot exceed 16K (16,384) for any CCP configuration. If DFF is configurated, the above sum cannot exceed 15,254 bytes. If DFF and MLTA is configured, the above sum cannot exceed 15,246 bytes less 3 bytes for each individual type of MLTA terminal. _____

Figure 10 (Part 6 of 6). Storage Requirements for Common Code and Control Tables – 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 1—General Applies to all CCP configurations		
Common code & control tables		Enter the total from Section 8 of Figure 4 here. ____
Communication support code	$CC4#2	Enter the $CC4#2 storage requirements from Figure 1 here. ____
BSCA trace	CCBS	If TRACEMLMP will be specified at startup enter 650. ____
MLTA trace	CCML	If TRACEMLTA will be specified at startup enter 1325. ____
Section 1 total		Add results within this section, enter total here and in Section 3 below. ____

Figure 11 (Part 1 of 2). Storage Requirements for Resident CCP – 5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 2—Line buffers for BSCA communications lines. If no BSCALINE statements in assignment set, skip to Section 3.		
BSCA line buffers		Refer to TERMATTR, BSCALINE and BSCATERM statements within assignment set. Calculate the size of each line buffer as follows: Line Number 1 2 1 Enter (largest BLKL value used on each line) + 42. ____ ____ 2 If ITB-Y, TRANSP-N within TERMATTR statement, enter 1 here. ____ ____ 3 If ITB-Y, TRANSP-Y within TERMATTR statement, enter 7 here. ____ ____ 4 If DBLBUF-YES within BSCALINE statement, add lines 1, 2, and 3 and enter total here. ____ ____ 5 If XMCODE-ASCII specified in BSCALINE statement, add lines 1, 2, and 3 and enter total here. ____ ____ 6 If TYPE-SW within BSCALINE statement, enter number of IDEXSEND characters here. ____ ____ 7 If TYPE-SW within BSCALINE statement, enter number of IDEXRCV characters within BSCATERM statement here. ____ ____ 8 Add items 1 through 7 above for each BSCA line, and enter totals here. ____ ____
Section 2 total		Add totals of both BSCA lines, enter result here and in Section 3 below. ____ ____
Section 3—Resident CCP totals.		
Resident CCP total		Enter total from Section 1 here. ____ Enter total from Section 2 here. ____ Enter the size of the TP buffer from Figure 3 here. ____ Add the 3 lines above, and round up to the next 2K (2,048) increment. The difference is added to the size of TP buffer. The result represents the size of resident CCP and cannot exceed 48K (49,152). ____

Figure 11 (Part 2 of 2). Storage Requirements for Resident CCP—5704-SC1 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements	
Section 1—General Applies to all CCP configurations			
Transient areas	T1 & T2	Space reserved for CCP transients.	1536
Common CCP code	$CC4#1	Enter $CC4#1 storage requirements from Figure 8 here.	____
Program request count table	PCT	If PGMCNT-YES ($EFAC), enter 2 x (number of PROGRAM statements in assignment set).	____
User security	$CC4Z9	If SECURE-USER is used in $ESEC statement, enter size of user's security information (value of LUSI keyword).	____
Terminal attribute set	TAS	Enter 5 x (high sequence number specified in the ATTRID parameter of the TERMATTR assignment statement).	____
Section 1 total		Add results within this section, enter total here and in Section 10 below.	____
Section 2—Control information for MLTA communications lines. If no MLTALINE statements in assignment set, skip to Section 3.			
		Refer to MLTALINE and MLTATERM statements within assignment set.	
Define file/line control block	DTF/LCB	Enter 126 bytes for each MLTALINE statement.	MLTA Line Numbers . . . 1 2 3 4 5 6 7 8 — — — — — — — —
Terminal statistics tables	TST	Enter 5 x (number of MLTATERM statements) for each MLTALINE statement.	— — — — — — — —
Polling list (note 1)	POLLLIST	Enter 12 x (number of 1050 terminal entries in POLLLIST) for each MLTALINE statement.	— — — — — — — —
		Also enter 6 + (3 x number of non-1050 terminal entries in POLLLIST) for each MLTALINE statement.	— — — — — — — —
Check list		Enter 3 for each MLTALINE statement.	— — — — — — — —

Figure 12 (Part 1 of 6). Storage Requirements for Common Code and Control Tables – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
MLTA line buffers (note 1)		Enter the appropriate values calculated below for each terminal type on each MLTA line. 2741: MAXRECL + 15 = ____ 1050 or 2740 automatic polling: MAXRECL + 14 = ____ 1050 not automatic polling: MAXRECL + 17 = ____ 2740 control station, not automatic polling: MAXRECL + 17 = ____ 2740 transmit control: MAXRECL + 17 = ____ All others terminal types: MAXRECL + 15 = ____
Line totals		Add results above for each MLTA line. ____
Adaptor DTF	DTF	Main storage requirement for MLTA adaptor DTF. + 33
Section 2 total		Add line totals and adaptor requirements. Enter result here and in Section 10 below. ____ Note 1—The MLTA POLLLIST and line buffers are actually located just before $CC4#2.

Figure 12 (Part 2 of 6). Storage Requirements for Common Code and Control Tables — 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 3—Control information for BSCA and BSCC communications lines. If no BSCALINE statements within assignment set, skip to Section 4.		
		Line Number 1 2 3 4
BSCA line control block	DTF/LCB	Enter 137 bytes for each BSCALINE 1 or 2 ___ ___ If TYPE-SW within BSCALINE statement, enter 4 for those line numbers. — —
BSCC line control block	DTF CLB	Enter 132 bytes for each BSCALINE 3 or 4 ___ ___
Check list		Enter 3 for each BSCALINE 1 or 2 statement. — —
Line totals		Add results above for each BSCA line. ___ ___ ___ ___
Section 3 total		Add all line totals, enter result here and in Section 10 below. ___
Section 4—Control information for the serial I/O channel and BSCA or BSCC PORTLINE. If no PORTLINE statement within assignment set, skip to Section 5.		
SIOC line control block	DTF/CLB	Enter 131 bytes for each PORTLINE type SIOC. ___
BSCA PORTLINE control block	DTF/LCB	Enter 137 bytes for each PORTLINE type line 1, line 2, or TTASK. ___
BSCC PORTLINE control block	DTF/CLB	Enter 132 bytes for each PORTLINE type line 3 or line 4. ___
Check list		Enter 3 for each PORTLINE type line 1, line 2, or TTASK. ___
Section 4 total		Add all line totals; enter result here and in section 10 below. ___

Figure 12 (Part 3 of 6). Storage Requirements for Common Code and Control Tables – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 5--Control information for BSCA, BSCC and MLTA terminals. Control information for SIOC ports.		
Terminal unit block	TUB	Refer to MLTATERM, BSCATERM, and PORTLINE statements within assignment set. Enter 39 x (number of MLTA and BSCA and BSCC terminals). ____ Also enter 19 x (number of BSCA switched terminals) +1. ____ Also enter 19 x (number of command terminals) + 1. ____ Also enter 58 x (number of acquirable and nonacquirable BSCA or BSCC ports) ____ Also enter 39 x (number of acquirable and non-acquirable SIOC ports). ____ Also enter 45 x (number of task chaining terminals as specified on MAXCHAIN parameter in the SYSTEM assignment statement). ____
Section 5 total		Add results within this section, enter total here and in Section 10 below. ____
Section 6—Control information for BSCA, BSCC, and MLTA terminals. Control information for SIOC ports. If terminal name table MOVEOUT (MOVTNT) specified, skip to Section 7.		
Terminal name table	TNT	Refer to TERMNAME and PORTLINE statements within assignment set. Enter 23 + (11 x number of TERMNAME statements). ____ Enter 11 + (11 x number of SIOC ports, BSCA ports, and BSCC ports specified). ____
Section 6 total		Add results within this section, enter total here and in Section 10 below. ____

Figure 12 (Part 4 of 6). Storage Requirements for Common Code and Control Tables – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 7—Control information for symbolic and associated symbolic files. If resident open/close (RESOPN) specified or more than 50 disk files or resident DFF indexed (DFFINDX) specified, skip to Section 9. If no SYMFILE statements within assignment set, skip to Section 8.		
File specification blocks	FSB	Refer to SYMFILE, BSCATERM, and program statements within assignment set. Enter (number of SYMFILES) x (number of command terminals) + (number of user programs allowed to run concurrently*). * 15 maximum _____
Symbolic file table Section 7 total	XDT	Calculate the size of one XDT for each SYMFILE statement. Each table = 9 + number of diskfile references within SYMFILE statement. Enter total for all SYMFILE statements. The XDT is acually located following the SDF address list below. Add results within this section, enter total here and in Section 10 below. _____
Section 8—Control information for disk files. If resident open/close (RESOPN) specified or more than fifty disk files or resident DFF indexes (DFFINDX) specified, skip to Section 9.		
Share DTF address list	SDF @ List	Enter 12 x (number of physical disk files). Also enter value in Section 10 below. _____
Section 9—Control information for user programs.		
Program characteristics table	PCT	Enter 6 + the number of sectors of PCT in $CCPFILE, or as a rough approximation, enter 1.5 x number of program statements rounded up to the nearest multiple of 6. Also enter value in Section 10 below. _____

Figure 12 (Part 5 of 6). Storage Requirements for Common Code and Control Tables — 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 10—CCP common code and control table totals.		
		Enter total from Section 1 here. ____
		Enter total from Section 2 here. ____
		Enter total from Section 3 here. ____
		Enter total from Section 4 here. ____
		Enter total from Section 5 here. ____
		Enter total from Section 6 here. ____
		Enter total from Section 7 here. ____
		Enter total from Section 8 here. ____
		Enter total from Section 9 here. ____
		If DFF is configured without DFF moved out, enter 3900 here. ____
		Enter 21 for line address holder. ____
Common code and control table total		Add all lines above. Sum represents size of all CCP common code and control tables. It cannot exceed 16K (16,384) for any CCP configuration. ____

Figure 12 (Part 6 of 6). Storage Requirements for Common Code and Control Tables — 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 1—General Applies to all CCP configurations		
Common code & control tables		Enter the total from Section 10 of Figure 12 here. ____
Communication support code	$CC4#2	Enter the $CC4#2 storage requirements from Figure 8 here. ____
BSCA trace	CCBS	If TRACEMLMP may be specified at startup enter 650. ____
MLTA trace	CCML	If TRACEMLTA may be specified at startup enter 1325. ____
Section 1 total		Add results within this section; enter total here and in Section 3 below. ____
Section 2—Line buffers for BSCA communications lines. If no BSCALINE statements in assignment set, skip to Section 3.		
BSCA line buffers		Refer to TERMATTR, BSCALINE and BSCATERM statements within assignment stage. Calculate the size of each line buffer as follows: Line 1 ____ Number 2 ____ 1 Enter (largest BLKL value used on each line rounded up to the nearest multiple of 256) + 42. 2 If ITB-Y, TRANSP-N within TERMATTR statement, enter 1 here. — — 3 If ITB-Y, TRANSP-Y within TERMATTR statement, enter 7 here. — — 4 If DBLBUF-YES within BSCALINE statement, add lines 1, 2, and 3 and enter total here. ____ ____ 5 If XMCODE-ASCII specified in BSCALINE statement, add lines 1, 2, and 3 and enter total here. ____ ____

Figure 13 (Part 1 of 6). Storage Requirements for Resident CCP – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
		6 If TYPE-SW within BSCALINE statement, enter number of IDEXSEND characters here. ____ ____
		7 If TYPE-SW within BSCALINE statement, enter number of IDEXRCV characters within BSCATERM statement here. ____ ____
Addressing list	ADDRLIST	8 If TYPE-CS in BSCALINE statement enter ((number of BSCATERM statements) x (number of ADDRCHAR characters + 3) + 1) for each line statement. ____ ____
Polling list	POLLLIST	9 If TYPE-CS in BSCALINE statement enter ((number of POLLLIST entries) x (number of POLLCHAR characters + 3) + 1) for each line statement. ____ ____
Terminal statistics tables	TST	10 Enter 4 + ((number of BSCATERM statements) x 2 x (number of POLLCHAR + 7)) for each line statement. ____ ____
		11 Add items 1 through 10 above for each BSCA line, and enter totals here. ____ ____
Section 2 total		Add totals of both BSCA lines, enter result here and in Section 3 below. ____
Section 3—Resident CCP sub-total.		
		Enter total from Section 1 here. ____
		Enter total from Section 2 here. ____
		Enter the size of the TP buffer from Figure 9 here. ____
Resident CCP sub-total		Add the 3 lines above, and round up to the next 2K (2,048) increment. The difference is added to the size of TP buffer. The result cannot exceed 48K (49,152). Enter result here. ____
		For each optional DFF buffer (maximum of one per BSCA line), add 2048 to total above. Enter here and in Section 12 below. ____

Figure 13 (Part 2 of 6). Storage Requirements for Resident CCP – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements	
Section 4—Moved out terminal name tables. If terminal name table MOVEOUT (MOVTNT) not specified, skip to Section 5.			
Terminal name table	TNT	Refer to MLTATERM, BSCATERM, and TERMNAME statements within assignment set. Enter 23 + (11 x number of TERMNAME statements). Enter 11 + (11 x number of SIOC ports and BSCA ports or BSCC ports specified. Add results in this section; enter total here and in Section 12 below.	_____ _____ _____
Section 5—Resident pseudo OPEN/CLOSE of disk files. If resident open/close (RESOPN) not specified, skip to Section 6.			
Resident open/close	$CCROC	Resident pseudo OPEN-CLOSE requires 10K (10,240). Also enter value in Section 12 below.	10,240
Section 6—Control information within the external pointer list (EPL). If resident open/close (RESOPN) not specified and not more than fifty disk files (DISKFILE) specified and resident DFF indexes (DFFINDX) not specified, skip to Section 7.			
Header	EPL	Contains four 2 byte pointers.	+8
Symbolic file table	XDT	Calculate the size of one table for each SYMFILE statement. Each table = 9 + number of diskfile references within SYMFILE statement. Enter total for all SYMFILE statements. The XDT is actually located following the SDF address list below.	_____
File specification blocks	FSB	Enter (number of SYMFILES) x ((number of command terminals) + (number of user programs allowed to run concurrently)).	_____
Share DTF address list	SDF @ List	Enter 12 x (number of physical disk files).	_____
Object library address table	EP@QCS	Contains all referenced Q codes and their cylinder/sector addresses that were in the assignment set.	_____
DFF indexes		If DFFINDX-YES is specified, enter 6 x (number of DFF formats used).	_____

Figure 13 (Part 3 of 6). Storage Requirements for Resident CCP – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements	
Section 6 total		Add results within this section, and round up to the next 2K (2,048) increment. If result is greater than 4K (4,096), then change it to 4K. Enter result here and in Section 12 below. The difference between the space used for disk files and the final result is used for DFF resident indexes. However all of the format indexes may not be in main storage even if DFFINDX-YES is specified.	____
Section 7—Resident program request. If resident program request (RESREQ) not specified, skip to Section 8.			
Resident program request	$CCRPR	Resident program request requires 4K (4,096). Also enter value in Section 12 below.	4096
Section 8—Display Format facility MOVEOUT. If DFF MOVEOUT (MOVDFF) not specified, skip to Section 9.			
DFF MOVEOUT	$CC4DF	DFF requires 4K (4,096). Also enter value in Section 12 below.	4096
Section 9—Control information for BSCC. If not using BSCC (BSCA lines 3 or 4), skip to Section 12.			
Transient area	T3 + $CC4U0	Space reserved for BSCC transients and BSCC system message formatting.	2048
Communication support code	$CC4#M	Enter the $CC4#M storage requirements from Figure 8 here.	____
BSCC trace	$$BSYT	If TRACEBSCC will be specified at startup enter 256 x (1 + BSCCBLK)	____
Addressing list	ADDRLIST	If TYPE-CS in BSCALINE statement, enter ((number of BSCATERM statements) x (number of ADDRCHAR characters + 3) + 1) for each line statement.	Line 3 ____ Number 4 ____
Polling list	POLLLIST	If TYPE-CS in BSCALINE statement, enter [(number of POLLLIST entries) x (number of POLLCHAR characters + 3) + 1] for each line statement.	____ ____
Terminal statistics tables	TST	Enter 4 + ((number of BSCATERM statements) x 2 x (number of POLLCHAR + 7)) for each line statement.	____ ____
BSCC work area		BSCC work area requirements enter 206 for each BSCC line used.	___ ___
		Add totals of both BSCC lines, enter result	____

Figure 13 (Part 4 of 6). Storage Requirements for Resident CCP — 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements	
BSCC line buffers		Calculate the size of each line buffer as follows: Line 3 / Number 4	
		1 Enter (largest BLKL value used on each line rounded up to the nearest multiple of 256) + 42. ____ ____	
		2 If ITB-Y, TRANSP-N within TERMATTR statement, enter 1 here. ___ ___	
		3 If ITB-Y, TRANSP-Y within TERMATTR statement, enter 7 here. ___ ___	
		4 If DBLBUF-YES within BSCALINE statement, add lines 1, 2, and 3 and enter total here. ____ ____	
		5 If TYPE-SW within BSCALINE statement, enter number of IDEXSEND characters here. ____ ____	
		6 If TYPE-SW within BSCALINE statement, enter number of IDEXRCV characters within BSCATERM statement here. ____ ____	
		7 Add items 1 through 6 above for each BSCC line, and enter totals here. ____ ____	
		Add buffer totals of both BSCC lines, and enter result.	____
Section 9 total		Add results within this section, and round up to the next 2K (2,048) increment. Enter result here and in Section 12.	____
Section 10—Control information and support for SIOC. If SIOC not specified, skip to Section 12.			
Communication support code	$CC4#S	SIOC requires 4K (4,096). Also enter value in Section 12 below.	4096
Section 11—Line buffers for SIOC and BSCA or BSCC PORTLINE communications. If no PORTLINE statement in assignment set, skip to Section 12.			
SIOC line buffer		Refer to the PORTLINE statement within assignment set. Enter (BLKL value used on PORTLINE statement rounded up to the nearest multiple of 256) + 42. Also enter value in Section 12.	____

Figure 13 (Part 5 of 6). Storage Requirements for Resident CCP – 5704-SC2 only

CCP Components	Internal Terminology	Details for Estimating Storage Requirements
Section 12—Resident CCP totals.		
		Enter resident CCP sub-total from Section 3 here. ____
		Enter total from Section 4 here. ____
		Enter total from Section 5 here. ____
		Enter total from Section 6 here. ____
		Enter total from Section 7 here. ____
		Enter total from Section 8 here. ____
		Enter total from Section 9 here. ____
		Enter total from Section 10 here. ____
		Enter total from Section 11 here. ____
Resident CCP total		Add the lines above. The result represents the total main storage requirement of resident CCP. ____

Figure 13 (Part 6 of 6). Storage Requirements for Resident CCP – 5704-SC2 only

For any currently executing user program using DFF, the following storage is appended to the user program.

Program Appended Storage

Use the following to calculate the size of the terminal table and the format table.

Enter 37 x (value of DFFMTERM within PROGRAM statement) _____

Enter 18 x (value of DFFNDF within PROGRAM statement) _____

Add 127 + 127

Add the three lines above, round up to the next 256 bytes and enter result here. _____

The minimum storage size of 256 for these tables will contain enough space for any one of the following combinations of terminals and formats.

1 terminal + (1 to 5 formats) or
2 terminals + (1 to 3 formats) or
3 terminals + 1 format

Enter the size specified for the field descriptor table (DFFSFDT within PROGRAM statement) rounded up to the next 256 byte multiple. _____

The decimal length of a field descriptor table can be estimated as 256 bytes for the first 1 to 17 fields defined within the DFF format, and another 256 bytes for each 18 additional fields or fraction thereof.

Figure 14. DFF Storage Requirements Within the User Program Area

Enter the size of resident CCP from Figure 11 or Figure 13 here. _____

Enter your estimated requirement for the size of the user program area here. Take into account the DFF storage requirements from Figure 14 above. Take into account the storage requirements for memory resident overlays (MORCOR parameter on system statement at assignment). Also take into account any storage required for use to external buffers. Use the value obtained for the value of the MINUPA parameter within system statement at assignment. _____

Add above figures and enter result here. Result represents the partition size required for the executable CCP configuration. _____

Figure 15. Executable CCP Partition Size Requirement

MAXIMUM CCP SYSTEM EXAMPLE (5704-SC2 ONLY)

This section shows an example of a maximum CCP system in which a 512K System/3 Model 15D with 4 BSC lines is used. This example supports 120 terminals, 192 files, and 432 programs, and it illustrates how main storage is allocated for a large configuration.

Figure 16 shows a detailed listing of the main storage requirements for this example. The resident CCP total of 94,921 bytes can be achieved because as many routines and buffers as possible have been mapped out to other parts of main storage via generation and assignment options; normally, resident CCP code is limited to 64K bytes.

As supporting documentation, Figure 17 shows the generation options selected for this example and Figure 18 shows the assignment set.

CCP Components	Internal Terminology	Details for Estimating Storage Requirement		
Section 1—General				
Transient areas	T1 & T2	Space reserved for CCP transients.		1536
Common CCP code	$CC4#1	Enter $CC4#1 storage requirements.		7164
Terminal attribute set	TAS	Enter 5 x (high ATTRID on the TERMATTR statement)		20
Section 1 total		Add results in this section.		8720
Section 3—Control information for BSCA & BSCC lines				
BSCA line control block	DTF/LCB	Enter 137 bytes for each BSCALINE 1 or 2.	137	137
BSCC line control block	DTF/CLB	Enter 132 bytes for each BSCALINE 3 or 4.	132	132
Check list		Enter 3 for each line 1 & 2.	3	3
Line totals		Add results for each line.	272	272
Section 3 total		Add results in this section.		544
Section 5—Control information for terminals				
Terminal unit block	TUB	Enter 39 x (number of terminals).		4680
		Enter 19 x (number of command terminals).		1710
		Enter 45 x (number of task-chaining terminals).		231
Section 5 total		Add results in this section.		6621
Section 9—Control information for user programs				
Program control table	PCT	Enter 6 + 6 x (number of PCT sectors in $CCPFILE).		462

Figure 16 (Part 1 of 5). Storage Requirements for Resident CCP (Example)—5704-SC2 Only

CCP Components	Internal Terminology	Details for Estimating Storage Requirement	
Section 10--CCP common code and control table totals			
		Enter Section 1 total.	8720
		Enter Section 3 total.	544
		Enter Section 5 total.	6621
		Enter Section 9 total.	462
		Enter 21 for ADDR. holder	21
Common code total		Add all lines above. Cannot exceed 16K (16,384).	16368

Figure 16 (Part 2 of 5). Storage Requirements for Resident CCP (Example)—5704-SC2 Only

CCP Components	Internal Terminology	Details for Estimating Storage Requirement		
Section 1—General				
Common code total		Enter total from Section 10.		16368
Communication code	$CC4#2	Enter the $CC4#2 storage requirements.		15678
Section 1 total		Add results in this section.		32046
Section 2—Line buffers for BSCA lines				
BSCA line buffers		Calculate the size of each line buffer as follows:		
		1 Enter (largest BLKL used on each line) + 42.	2090	2090
Addressing list	ADDRLIST	Enter ((number of BSCATERMS) x (number of ADDRCHAR + 3) + 1) for each line.	211	211
Polling list	POLLLIST	Enter ((number of POLLLIST entries) x (number of POLLCHAR + 3) + 1) for each line.	211	211
Terminal statistics table	TST	Enter 4 + ((number of BSCATERMS) x 2 x (number of POLLCHAR + 7)) for each line.	664	664
Line totals		Add results for each line.	3176	3176
Section 2 total		Add results in this section.		6352
Section 3—Resident CCP sub-total				
		Enter Section 1 total.		32046
		Enter Section 2 total.		6352
		Enter TP buffer size.		10736
Resident CCP sub-total		Add all lines above. Round up to the next 2K (2,048) increment. Difference is added to the size of TP buffer. Cannot exceed 48K (49,152).		49134

Figure 16 (Part 3 of 5). Storage Requirements for Resident CCP (Example)—5704-SC2 Only

CCP Components	Internal Terminology	Details for Estimating Storage Requirement	
Section 4—Moved out terminal name tables			
Terminal name table	TNT	Enter 23 + (11 x number TERMNAME statements) rounded up to next 2K.	2048
Section 5—Resident pseudo open/close of disk files			
Resident open/close	$CCROC	Resident pseudo open/close requires 10K (10,240).	10240
Section 6—Control information within the external pointer list			
Header	EPL	Contains 4 two byte ptrs.	8
Symbolic file table	XDT	Enter 9 + (number of disk file ref. within each SYMFILE statement).	90
File specification blocks	FSB	Enter (# of SYMFILES) x ((# command terminals) + (# user TCBS))	945
Share DTF address list	SDF@ list	Enter 12 x (number of physical disk files).	2304
Object library address table	EP@QCS	All referenced Q-codes and C/S addresses.	0
DFF indexes		Enter 6 x (number of DFF formats used).	576
Section 7 total		Add results in this section. Round up to next 2K (2,048) increment.	4096
Section 7—Resident program request			
Resident program request	$CCRPR	Resident program request requires 4K (4,096).	4096
Section 8—Display format facility MOVEOUT			
DFF MOVEOUT	$CC4DF	DFF requires 4K (4,096).	4096

Figure 16 (Part 4 of 5). Storage Requirements for Resident CCP (Example)—5704-SC2 Only

CCP Components	Internal Terminology	Details for Estimating Storage Requirement		
Section 9—Control information for BSCC				
Transient area	T3 & $CC4U0	Space reserved for BSCC transients 2K (2,048).		2048
Communication support area	$CC4#M	Enter the $CC4#M storage requirements.		13935
Addressing list	ADDRLIST	Enter (number of BSCATERMS) x (number of ADDRCHAR + 3) + 1 for each line.	211	211
Polling list	POLLLIST	Enter (number of POLLLIST entries) x (number of POLLCHAR + 3) + 1 for each line.	211	211
Terminal statistics table	TST	Enter 4 + ((number of BSCATERMS) x 2 x (number of POLLCHAR + 7)) for each line.	664	664
BSCC work area		Enter 206 for each line.	206	206
BSCC line buffers		Calculate the size of each BSCC line as follows:		
		1 Enter (largest BLKL used on each line) + 42.	1322	1322
Line totals		Add results for each line.	2614	2614
Section 10 total		Add results in this section.		21211
Section 12—Resident CCP totals				
		Enter resident CCP subtotal from Section 3.		49134
		Enter Section 4 total.		2048
		Enter Section 5 total.		10240
		Enter Section 6 total.		4096
		Enter Section 7 total.		4096
		Enter Section 8 total.		4096
		Enter Section 9 total.		21211
Resident CCP total		Add the lines above. The result represents the total main storage requirement of resident CCP.		94921

Figure 16 (Part 5 of 5). Storage Requirements for Resident CCP (Example)—5704-SC2 Only

```
*                GENERATION USED FOR A MAXIMUM CCP SYSTEM
*                          (5704-SC2 ONLY)
*
************************************************************************ 00010000
*      G E N    U S E D    F O R    A    M A X I M U M    S E T       *
************************************************************************ 00030000
// CALL $CG1G1,R1
// RUN
        $EFAC ESCAPE-'//////',    -- NO  / X'XXXXXXXXXXX' -- ESCAPE MODEX00040000
              PGMCNT-NO,          -- NO -- PROGRAM USE COUNT              X00050000
              FORMAT-YES,         -- NO -- DFF SUPPORT                    X00060000
              PRUF-YES,           -- NO -- PRUF SUPPORT                   X00070000
              ACCEPT-YES,         -- NO -- RESIDENT INPUT DATA HANDLER    X00080000
              RESOPN-YES,         -- NO -- RESIDENT OPEN CLOSE            X00090000
              BSYPRT-YES,         -- NO -- BUSY PRINTER HANDLING          X00100000
              MOVTNT-YES,         -- NO -- TNT MOVED OUT BY STARTUP       X00110000
              MOVDFF-YES,         -- NO -- DFF MOD. MOVED OUT BY STARTUP X00120000
              RESREQ-YES,         -- NO -- RESIDENT PROGRAM REQUEST       X00130000
              CPUMSG-YES,         -- NO -- S TYPE MSG'S (FOR S/34 CPU)    X00160000
              TTASK-YES           -- NO -- TASK-TO-TASK SUPPORT            00180000
        $EPLG LANG-COBOL,         -- RPGII / FORTRAN / ASSEM --           X00190000
              PPUNIT-R2           -- R1 / F1 / F2 --                       00200000
        $EPLG LANG-FORTRAN,       -- COBOL / RPGII / ASSEM --             X00210000
              PPUNIT-R2           -- R1 / F1 / F2 --                       00220000
        $EPLG LANG-ASSEM,         -- COBOL / RPGII / FORTRAN --           X00230000
              PPUNIT-R2           -- R1 / F1 / F2 --                       00240000
        $EPLG LANG-RPGII,         -- COBOL / FORTRAN / ASSEM --           X00250000
              PPUNIT-R2           -- R1 / F1 / F2 --                       00260000
        $ESEC SECURE-CCP          -- NO / USER (REQUIRES 'LUSI' PARM) --   00270000
        $EFIL SETS-12,            -- 1 - 25 -- MAX # OF SETS ON FILE      X00280000
              PROGS-10,           -- 1 - 999 -- # OF PROGS/SET            X00290000
              DFILES-5,           -- 1 - 192 -- # OF DISK FILES/SET       X00300000
              TERMS-8,            -- 2 - 254 -- # OF TERMINALS/SET       X00310000
              DUMPS-1,            -- 2 - 99 --  # OF DUMP AREAS WANTED   X00320000
              CORE-512K,          -- 96K/128K/160K/192K/224K/256K/384K    X00330000
              FLPACK-F2F2F2,      -- NAME OF PACK --$CCPFILE'S PACK NAME X00340000
              FLUNIT-F2,          -- R1 / F1 / R2 --$CCPFILE'S UNIT NAME X00350000
              DPTRAC-10           -- # OF TRACE AREA TRACKS
        $EBSC BSCA-2,             -- 0 - 2 -- # OF BSCA LINES             X00370000
              DA-YES,             -- NO -- DISPLAY ADAPTER SUPPORT        X00380000
              CS-YES,             -- NO -- BSCA CONTROL STATION SUPPORT   X00420000
              GETMSG-YES,         -- NO -- GET A MESSAGE                  X00430000
              RECSEP-1E,          -- TWO HEX DIGITS -- RECORD SEPARATER   X00450000
              RESPOL-YES,         -- NO -- RESIDENT BSCA POLLING          X00490000
              INTPOL-YES,         -- NO -- INTERVAL POLLING               X00500000
              EBCDIC-YES,         -- NO -- EBCDIC TRANSMISSION CODE
              PORT-YES            -- NO -- BSCA PORTLINE SUPPORT           00520000
        $EBSD TYPE-3275M1         -- SEE SYSTEM REFERENCE MANUAL --        00530000
        $EBSD TYPE-3277M1         -- SEE SYSTEM REFERENCE MANUAL --        00540000
        $EBSD TYPE-3284M1         -- SEE SYSTEM REFERENCE MANUAL --        00550000
        $EBSD TYPE-3286M1         -- SEE SYSTEM REFERENCE MANUAL --
        $EBSD TYPE-3275M2         -- SEE SYSTEM REFERENCE MANUAL --        00570000
        $EBSD TYPE-3277M2         -- SEE SYSTEM REFERENCE MANUAL --        00580000
        $EBSD TYPE-3284M2         -- SEE SYSTEM REFERENCE MANUAL --        00590000
        $EBSD TYPE-3286M2         -- SEE SYSTEM REFERENCE MANUAL --        00600000
        $ECSC BSCC-2,             -- 0 - 2 --# OF BSCC LINES              X00640000
              GETMSG-YES,         -- NO -- GET A MESSAGE                  X00650000
              RECSEP-1E,          -- TWO HEX DIGITS --RECORD SEPARATOR    X00670000
              INTPOL-YES,         -- NO -- INTERVAL POLLING               X00710000
              PORT-YES,           -- NO -- BSCC PORTLINE SUPPORT          X00720000
              EBCDIC-YES,         -- NO -- EBCDIC TRANSMISSION CODE
              CS-YES              -- NO -- BSCC CONTROL STATION SUPPORT    00740000
        $ECSD TYPE-3275M1         -- SEE SYSTEM REFERENCE MANUAL --        00750000
        $ECSD TYPE-3277M1         -- SEE SYSTEM REFERENCE MANUAL --        00760000
        $ECSD TYPE-3284M1         -- SEE SYSTEM REFERENCE MANUAL --        00770000
        $ECSD TYPE-3286M1         -- SEE SYSTEM REFERENCE MANUAL --
        $ECSD TYPE-3275M2         -- SEE SYSTEM REFERENCE MANUAL --        00790000
        $ECSD TYPE-3277M2         -- SEE SYSTEM REFERENCE MANUAL --        00800000
        $ECSD TYPE-3284M2         -- SEE SYSTEM REFERENCE MANUAL --        00810000
        $ECSD TYPE-3286M2         -- SEE SYSTEM REFERENCE MANUAL --        00820000
        $EGEN DSUNIT-F1,          -- R1 -- IPL UNIT                       X00860000
              CCUNIT-R2,          -- R1 / F1 / F2 --EXECUTION UNIT        X00870000
              WKUNIT-D2,          -- 'UNIT,UNIT,UNIT' --UNIT FOR WORK     X00880000
              WKPACK-D2D2D2,      -- 'PACK,PACK,PACK' --PACKNAME FOR WRK X00890000
              DIUNIT-R1,          -- F1 / R2 / F2 -- CCP PID              X00900000
              MINRES-NO           -- MINIMUM RESIDENT CODE OPTION
              CARD-NO             -- YES --CARDLESS CCP GENERATION        X00920000
              DPPACK-D2D2D2       -- $CCPDUMP FILE'S PACK NAME
              DPUNIT-D2           -- $CCPDUMP FILE'S UNIT
```

Figure 17. CCP Generation Listing Used for Example—5704-SC2 Only

```
*
*  EXAMPLE - ASSIGNMENT SET LISTING OF A MAXIMUM CCP SYSTEM
*
*
*
// SET ID-7,ACTION-REPLACE
*
// SYSTEM MINUPA-80K,MINTPBUF-10000,PGMREQL-79,COMMANDL-80,
//        MAXCHAIN-5,POLTIME-NO,PASSWORD-SHALOM,DFFPACK-PROGRAM
*
// TERMATTR ATTRID-01,TRANSLAT-YES,UPCASE-YES,SWITCHED-NO,BLKL-2048,
//          DATAFORM-MESSAGE,DFF3270-YES
*
// TERMATTR ATTRID-02,TRANSLAT-YES,UPCASE-YES,SWITCHED-NO,BLKL-1792,
//          DATAFORM-MESSAGE,DFF3270-NO
*
// TERMATTR ATTRID-03,TRANSLAT-YES,UPCASE-YES,BLKL-1280,
//          DATAFORM-MESSAGE,DFF3270-YES
*
// TERMATTR ATTRID-04,TRANSLAT-YES,UPCASE-YES,BLKL-817,
//          DATAFORM-MESSAGE
*
// BSCALINE TYPE-CS,LINENUM-1,
// POLLLIST-'A0,A1,A2,A3,A4,A5,A6,A7,A8,A9,
//           B0,B1,B2,B3,B4,B5,B6,B7,B8,B9,
//           C0,C1,C2,C3,C4,C5,C6,C7,C8,C9'
*
// BSCATERM TERMID-A0,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*60604040*,POLLCHAR-*40404040*
// BSCATERM TERMID-A1,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*
// BSCATERM TERMID-A2,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C2C2*,POLLCHAR-*4040C2C2*
// BSCATERM TERMID-A3,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C3C3*,POLLCHAR-*4040C3C3*
// BSCATERM TERMID-A4,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C4C4*,POLLCHAR-*4040C4C4*
// BSCATERM TERMID-A5,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C5C5*,POLLCHAR-*4040C5C5*
// BSCATERM TERMID-A6,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C6C6*,POLLCHAR-*4040C6C6*
// BSCATERM TERMID-A7,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C7C7*,POLLCHAR-*4040C7C7*
// BSCATERM TERMID-A8,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C8C8*,POLLCHAR-*4040C8C8*
// BSCATERM TERMID-A9,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C9C9*,POLLCHAR-*4040C9C9*
// BSCATERM TERMID-B0,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D1D1*,POLLCHAR-*4040D1D1*
// BSCATERM TERMID-B1,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D2D2*,POLLCHAR-*4040D2D2*
// BSCATERM TERMID-B2,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D3D3*,POLLCHAR-*4040D3D3*
// BSCATERM TERMID-B3,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D4D4*,POLLCHAR-*4040D4D4*
// BSCATERM TERMID-B4,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D5D5*,POLLCHAR-*4040D5D5*
// BSCATERM TERMID-B5,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D6D6*,POLLCHAR-*4040D6D6*
// BSCATERM TERMID-B6,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D7D7*,POLLCHAR-*4040D7D7*
// BSCATERM TERMID-B7,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D8D8*,POLLCHAR-*4040D8D8*
// BSCATERM TERMID-B8,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D9D9*,POLLCHAR-*4040D9D9*
// BSCATERM TERMID-B9,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E2E2*,POLLCHAR-*4040E2E2*
// BSCATERM TERMID-C0,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E3E3*,POLLCHAR-*4040E3E3*
// BSCATERM TERMID-C1,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E4E4*,POLLCHAR-*4040E4E4*
// BSCATERM TERMID-C2,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E5E5*,POLLCHAR-*4040E5E5*
// BSCATERM TERMID-C3,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E6E6*,POLLCHAR-*4040E6E6*
// BSCATERM TERMID-C4,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E7E7*,POLLCHAR-*4040E7E7*
// BSCATERM TERMID-C5,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E8E8*,POLLCHAR-*4040E8E8*
// BSCATERM TERMID-C6,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E9E9*,POLLCHAR-*4040E9E9*
// BSCATERM TERMID-C7,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F1F1*,POLLCHAR-*4040F1F1*
// BSCATERM TERMID-C8,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F2F2*,POLLCHAR-*4040F2F2*
// BSCATERM TERMID-C9,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F3F3*,POLLCHAR-*4040F3F3*
*
// BSCALINE TYPE-CS,LINENUM-2,
// POLLLIST-'H0,H1,H2,H3,H4,H5,H6,H7,H8,H9,
//           I0,I1,I2,I3,I4,I5,I6,I7,I8,I9,
//           J0,J1,J2,J3,J4,J5,J6,J7,J8,J9'
*
// BSCATERM TERMID-H0,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*60604040*,POLLCHAR-*40404040*
// BSCATERM TERMID-H1,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*
// BSCATERM TERMID-H2,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C2C2*,POLLCHAR-*4040C2C2*
// BSCATERM TERMID-H3,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C3C3*,POLLCHAR-*4040C3C3*
// BSCATERM TERMID-H4,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C4C4*,POLLCHAR-*4040C4C4*
// BSCATERM TERMID-H5,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C5C5*,POLLCHAR-*4040C5C5*
// BSCATERM TERMID-H6,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C6C6*,POLLCHAR-*4040C6C6*
// BSCATERM TERMID-H7,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C7C7*,POLLCHAR-*4040C7C7*
// BSCATERM TERMID-H8,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C8C8*,POLLCHAR-*4040C8C8*
// BSCATERM TERMID-H9,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C9C9*,POLLCHAR-*4040C9C9*
// BSCATERM TERMID-I0,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D1D1*,POLLCHAR-*4040D1D1*
// BSCATERM TERMID-I1,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D2D2*,POLLCHAR-*4040D2D2*
// BSCATERM TERMID-I2,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D3D3*,POLLCHAR-*4040D3D3*
// BSCATERM TERMID-I3,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D4D4*,POLLCHAR-*4040D4D4*
// BSCATERM TERMID-I4,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D5D5*,POLLCHAR-*4040D5D5*
// BSCATERM TERMID-I5,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D6D6*,POLLCHAR-*4040D6D6*
// BSCATERM TERMID-I6,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D7D7*,POLLCHAR-*4040D7D7*
// BSCATERM TERMID-I7,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D8D8*,POLLCHAR-*4040D8D8*
// BSCATERM TERMID-I8,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D9D9*,POLLCHAR-*4040D9D9*
// BSCATERM TERMID-I9,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E2E2*,POLLCHAR-*4040E2E2*
// BSCATERM TERMID-J0,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E3E3*,POLLCHAR-*4040E3E3*
// BSCATERM TERMID-J1,
//          TYPE-3277M2,ATTRID-'1,2',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E4E4*,POLLCHAR-*4040E4E4*
// BSCATERM TERMID-J2,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E5E5*,POLLCHAR-*4040E5E5*
// BSCATERM TERMID-J3,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E6E6*,POLLCHAR-*4040E6E6*
// BSCATERM TERMID-J4,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E7E7*,POLLCHAR-*4040E7E7*
// BSCATERM TERMID-J5,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E8E8*,POLLCHAR-*4040E8E8*
// BSCATERM TERMID-J6,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E9E9*,POLLCHAR-*4040E9E9*
// BSCATERM TERMID-J7,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F1F1*,POLLCHAR-*4040F1F1*
// BSCATERM TERMID-J8,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F2F2*,POLLCHAR-*4040F2F2*
// BSCATERM TERMID-J9,
//          TYPE-3286M2,ATTRID-'1,2',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F3F3*,POLLCHAR-*4040F3F3*
*
// BSCALINE TYPE-CS,LINENUM-3,
```

Figure 18 (Part 1 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// POLLLIST='M0,M1,M2,M3,M4,M5,M6,M7,M8,M9,
//          N0,N1,N2,N3,N4,N5,N6,N7,N8,N9,
//          00,01,02,03,04,05,06,07,08,09'
*
// BSCATERM TERMID-M0,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*60604040*,POLLCHAR-*40404040*
// BSCATERM TERMID-M1,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*
// BSCATERM TERMID-M2,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C2C2*,POLLCHAR-*4040C2C2*
// BSCATERM TERMID-M3,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C3C3*,POLLCHAR-*4040C3C3*
// BSCATERM TERMID-M4,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C4C4*,POLLCHAR-*4040C4C4*
// BSCATERM TERMID-M5,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C5C5*,POLLCHAR-*4040C5C5*
// BSCATERM TERMID-M6,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C6C6*,POLLCHAR-*4040C6C6*
// BSCATERM TERMID-M7,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C7C7*,POLLCHAR-*4040C7C7*
// BSCATERM TERMID-M8,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C8C8*,POLLCHAR-*4040C8C8*
// BSCATERM TERMID-M9,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C9C9*,POLLCHAR-*4040C9C9*
// BSCATERM TERMID-N0,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D1D1*,POLLCHAR-*4040D1D1*
// BSCATERM TERMID-N1,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D2D2*,POLLCHAR-*4040D2D2*
// BSCATERM TERMID-N2,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D3D3*,POLLCHAR-*4040D3D3*
// BSCATERM TERMID-N3,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D4D4*,POLLCHAR-*4040D4D4*
// BSCATERM TERMID-N4,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D5D5*,POLLCHAR-*4040D5D5*
// BSCATERM TERMID-N5,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D6D6*,POLLCHAR-*4040D6D6*
// BSCATERM TERMID-N6,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D7D7*,POLLCHAR-*4040D7D7*
// BSCATERM TERMID-N7,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D8D8*,POLLCHAR-*4040D8D8*
// BSCATERM TERMID-N8,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D9D9*,POLLCHAR-*4040D9D9*
// BSCATERM TERMID-N9,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E2E2*,POLLCHAR-*4040E2E2*
// BSCATERM TERMID-O0,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E3E3*,POLLCHAR-*4040E3E3*
// BSCATERM TERMID-O1,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E4E4*,POLLCHAR-*4040E4E4*
// BSCATERM TERMID-O2,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E5E5*,POLLCHAR-*4040E5E5*
// BSCATERM TERMID-O3,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E6E6*,POLLCHAR-*4040E6E6*
// BSCATERM TERMID-O4,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E7E7*,POLLCHAR-*4040E7E7*
// BSCATERM TERMID-O5,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E8E8*,POLLCHAR-*4040E8E8*
// BSCATERM TERMID-O6,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E9E9*,POLLCHAR-*4040E9E9*
// BSCATERM TERMID-O7,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F1F1*,POLLCHAR-*4040F1F1*
// BSCATERM TERMID-O8,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F2F2*,POLLCHAR-*4040F2F2*
// BSCATERM TERMID-O9,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F3F3*,POLLCHAR-*4040F3F3*
*
// BSCALINE TYPE-CS,LINENUM-4,
// POLLLIST='T0,T1,T2,T3,T4,T5,T6,T7,T8,T9,
//           U0,U1,U2,U3,U4,U5,U6,U7,U8,U9,
//           V0,V1,V2,V3,V4,V5,V6,V7,V8,V9'
*
// BSCATERM TERMID-T0,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*60604040*,POLLCHAR-*40404040*
// BSCATERM TERMID-T1,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C1C1*,POLLCHAR-*4040C1C1*
// BSCATERM TERMID-T2,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C2C2*,POLLCHAR-*4040C2C2*
// BSCATERM TERMID-T3,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C3C3*,POLLCHAR-*4040C3C3*
// BSCATERM TERMID-T4,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C4C4*,POLLCHAR-*4040C4C4*
// BSCATERM TERMID-T5,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C5C5*,POLLCHAR-*4040C5C5*
// BSCATERM TERMID-T6,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C6C6*,POLLCHAR-*4040C6C6*
// BSCATERM TERMID-T7,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C7C7*,POLLCHAR-*4040C7C7*
// BSCATERM TERMID-T8,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C8C8*,POLLCHAR-*4040C8C8*
// BSCATERM TERMID-T9,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060C9C9*,POLLCHAR-*4040C9C9*
// BSCATERM TERMID-U0,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D1D1*,POLLCHAR-*4040D1D1*
// BSCATERM TERMID-U1,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D2D2*,POLLCHAR-*4040D2D2*
// BSCATERM TERMID-U2,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D3D3*,POLLCHAR-*4040D3D3*
// BSCATERM TERMID-U3,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D4D4*,POLLCHAR-*4040D4D4*
// BSCATERM TERMID-U4,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D5D5*,POLLCHAR-*4040D5D5*
// BSCATERM TERMID-U5,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D6D6*,POLLCHAR-*4040D6D6*
// BSCATERM TERMID-U6,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D7D7*,POLLCHAR-*4040D7D7*
// BSCATERM TERMID-U7,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D8D8*,POLLCHAR-*4040D8D8*
// BSCATERM TERMID-U8,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060D9D9*,POLLCHAR-*4040D9D9*
// BSCATERM TERMID-U9,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E2E2*,POLLCHAR-*4040E2E2*
// BSCATERM TERMID-V0,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E3E3*,POLLCHAR-*4040E3E3*
// BSCATERM TERMID-V1,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E4E4*,POLLCHAR-*4040E4E4*
// BSCATERM TERMID-V2,
//          TYPE-3277M2,ATTRID-'3,4',ONLINE-NO,COMMAND-YES,
//          ADDRCHAR-*6060E5E5*,POLLCHAR-*4040E5E5*
// BSCATERM TERMID-V3,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E6E6*,POLLCHAR-*4040E6E6*
// BSCATERM TERMID-V4,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E7E7*,POLLCHAR-*4040E7E7*
// BSCATERM TERMID-V5,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E8E8*,POLLCHAR-*4040E8E8*
// BSCATERM TERMID-V6,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060E9E9*,POLLCHAR-*4040E9E9*
// BSCATERM TERMID-V7,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F1F1*,POLLCHAR-*4040F1F1*
// BSCATERM TERMID-V8,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F2F2*,POLLCHAR-*4040F2F2*
// BSCATERM TERMID-V9,
//          TYPE-3286M2,ATTRID-'3,4',ONLINE-NO,COMMAND-NO,
//          ADDRCHAR-*6060F3F3*,POLLCHAR-*4040F3F3*
*
// TERMNAME NAME-TERMA0,TERMID-A0
// TERMNAME NAME-TERMA1,TERMID-A1
// TERMNAME NAME-TERMA2,TERMID-A2
// TERMNAME NAME-TERMA3,TERMID-A3
// TERMNAME NAME-TERMA4,TERMID-A4
// TERMNAME NAME-TERMA5,TERMID-A5
// TERMNAME NAME-TERMA6,TERMID-A6
// TERMNAME NAME-TERMA7,TERMID-A7
// TERMNAME NAME-TERMA8,TERMID-A8
// TERMNAME NAME-TERMA9,TERMID-A9
// TERMNAME NAME-TERMB0,TERMID-B0
// TERMNAME NAME-TERMB1,TERMID-B1
// TERMNAME NAME-TERMB2,TERMID-B2
// TERMNAME NAME-TERMB3,TERMID-B3
// TERMNAME NAME-TERMB4,TERMID-B4
// TERMNAME NAME-TERMB5,TERMID-B5
// TERMNAME NAME-TERMB6,TERMID-B6
// TERMNAME NAME-TERMB7,TERMID-B7
// TERMNAME NAME-TERMB8,TERMID-B8
// TERMNAME NAME-TERMB9,TERMID-B9
// TERMNAME NAME-TERMC0,TERMID-C0
```

Figure 18 (Part 2 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// TERMNAME NAME-TERMC1,TERMID-C1
// TERMNAME NAME-TERMC2,TERMID-C2
// TERMNAME NAME-TERMC3,TERMID-C3
// TERMNAME NAME-TERMC4,TERMID-C4
// TERMNAME NAME-TERMC5,TERMID-C5
// TERMNAME NAME-TERMC6,TERMID-C6
// TERMNAME NAME-TERMC7,TERMID-C7
// TERMNAME NAME-TERMC8,TERMID-C8
// TERMNAME NAME-TERMC9,TERMID-C9
// TERMNAME NAME-TERMH0,TERMID-H0
// TERMNAME NAME-TERMH1,TERMID-H1
// TERMNAME NAME-TERMH2,TERMID-H2
// TERMNAME NAME-TERMH3,TERMID-H3
// TERMNAME NAME-TERMH4,TERMID-H4
// TERMNAME NAME-TERMH5,TERMID-H5
// TERMNAME NAME-TERMH6,TERMID-H6
// TERMNAME NAME-TERMH7,TERMID-H7
// TERMNAME NAME-TERMH8,TERMID-H8
// TERMNAME NAME-TERMH9,TERMID-H9
// TERMNAME NAME-TERMI0,TERMID-I0
// TERMNAME NAME-TERMI1,TERMID-I1
// TERMNAME NAME-TERMI2,TERMID-I2
// TERMNAME NAME-TERMI3,TERMID-I3
// TERMNAME NAME-TERMI4,TERMID-I4
// TERMNAME NAME-TERMI5,TERMID-I5
// TERMNAME NAME-TERMI6,TERMID-I6
// TERMNAME NAME-TERMI7,TERMID-I7
// TERMNAME NAME-TERMI8,TERMID-I8
// TERMNAME NAME-TERMI9,TERMID-I9
// TERMNAME NAME-TERMJ0,TERMID-J0
// TERMNAME NAME-TERMJ1,TERMID-J1
// TERMNAME NAME-TERMJ2,TERMID-J2
// TERMNAME NAME-TERMJ3,TERMID-J3
// TERMNAME NAME-TERMJ4,TERMID-J4
// TERMNAME NAME-TERMJ5,TERMID-J5
// TERMNAME NAME-TERMJ6,TERMID-J6
// TERMNAME NAME-TERMJ7,TERMID-J7
// TERMNAME NAME-TERMJ8,TERMID-J8
// TERMNAME NAME-TERMJ9,TERMID-J9
// TERMNAME NAME-TERMM0,TERMID-M0
// TERMNAME NAME-TERMM1,TERMID-M1
// TERMNAME NAME-TERMM2,TERMID-M2
// TERMNAME NAME-TERMM3,TERMID-M3
// TERMNAME NAME-TERMM4,TERMID-M4
// TERMNAME NAME-TERMM5,TERMID-M5
// TERMNAME NAME-TERMM6,TERMID-M6
// TERMNAME NAME-TERMM7,TERMID-M7
// TERMNAME NAME-TERMM8,TERMID-M8
// TERMNAME NAME-TERMM9,TERMID-M9
// TERMNAME NAME-TERMN0,TERMID-N0
// TERMNAME NAME-TERMN1,TERMID-N1
// TERMNAME NAME-TERMN2,TERMID-N2
// TERMNAME NAME-TERMN3,TERMID-N3
// TERMNAME NAME-TERMN4,TERMID-N4
// TERMNAME NAME-TERMN5,TERMID-N5
// TERMNAME NAME-TERMN6,TERMID-N6
// TERMNAME NAME-TERMN7,TERMID-N7
// TERMNAME NAME-TERMN8,TERMID-N8
// TERMNAME NAME-TERMN9,TERMID-N9
// TERMNAME NAME-TERMO0,TERMID-O0
// TERMNAME NAME-TERMO1,TERMID-O1
// TERMNAME NAME-TERMO2,TERMID-O2
// TERMNAME NAME-TERMO3,TERMID-O3
// TERMNAME NAME-TERMO4,TERMID-O4
// TERMNAME NAME-TERMO5,TERMID-O5
// TERMNAME NAME-TERMO6,TERMID-O6
// TERMNAME NAME-TERMO7,TERMID-O7
// TERMNAME NAME-TERMO8,TERMID-O8
// TERMNAME NAME-TERMO9,TERMID-O9
// TERMNAME NAME-TERMT0,TERMID-T0
// TERMNAME NAME-TERMT1,TERMID-T1
// TERMNAME NAME-TERMT2,TERMID-T2
// TERMNAME NAME-TERMT3,TERMID-T3
// TERMNAME NAME-TERMT4,TERMID-T4
// TERMNAME NAME-TERMT5,TERMID-T5
// TERMNAME NAME-TERMT6,TERMID-T6
// TERMNAME NAME-TERMT7,TERMID-T7
// TERMNAME NAME-TERMT8,TERMID-T8
// TERMNAME NAME-TERMT9,TERMID-T9
// TERMNAME NAME-TERMU0,TERMID-U0
// TERMNAME NAME-TERMU1,TERMID-U1
// TERMNAME NAME-TERMU2,TERMID-U2
// TERMNAME NAME-TERMU3,TERMID-U3
// TERMNAME NAME-TERMU4,TERMID-U4
// TERMNAME NAME-TERMU5,TERMID-U5
// TERMNAME NAME-TERMU6,TERMID-U6
// TERMNAME NAME-TERMU7,TERMID-U7
// TERMNAME NAME-TERMU8,TERMID-U8
// TERMNAME NAME-TERMU9,TERMID-U9
// TERMNAME NAME-TERMV0,TERMID-V0
// TERMNAME NAME-TERMV1,TERMID-V1
// TERMNAME NAME-TERMV2,TERMID-V2
// TERMNAME NAME-TERMV3,TERMID-V3
// TERMNAME NAME-TERMV4,TERMID-V4
// TERMNAME NAME-TERMV5,TERMID-V5
// TERMNAME NAME-TERMV6,TERMID-V6
// TERMNAME NAME-TERMV7,TERMID-V7
// TERMNAME NAME-TERMV8,TERMID-V8
// TERMNAME NAME-TERMV9,TERMID-V9
*
// DISKFILE NAME-FIL000,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL001,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL002,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL003,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL004,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL005,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL006,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL007,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL008,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL009,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL010,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL011,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL012,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL013,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL014,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL015,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL016,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL017,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL018,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL019,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL020,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL021,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL022,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL023,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL024,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL025,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL026,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL027,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL028,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL029,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL030,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL031,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL032,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL033,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL034,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL035,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL036,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL037,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL038,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL039,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL040,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL041,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL042,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL043,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL044,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL045,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL046,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL047,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL048,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL049,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL050,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL051,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL052,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL053,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL054,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL055,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL056,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL057,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL058,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL059,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL060,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL061,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL062,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL063,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL064,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL065,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL066,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL067,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL068,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL069,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL070,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL071,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL072,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL073,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL074,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL075,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL076,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL077,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL078,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL079,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL080,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL081,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL082,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL083,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL084,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL085,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL086,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL087,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL088,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL089,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL090,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL091,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL092,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL093,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL094,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL095,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL097,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL098,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL099,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL100,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL101,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL102,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL103,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL104,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL105,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL106,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL107,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL108,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL109,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL110,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL111,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL112,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL113,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL114,ORG-I,RECL-16,KEYL-6,KEYPOS-11
```

Figure 18 (Part 3 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// DISKFILE NAME-FIL115,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL116,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL117,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL118,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL119,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL120,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL121,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL122,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL123,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL124,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL125,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL126,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL127,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL128,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL129,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL130,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL131,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL132,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL133,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL134,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL135,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL136,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL137,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL138,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL139,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL140,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL141,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL142,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL143,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL144,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL145,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL146,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL147,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL148,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL149,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL150,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL151,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL152,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL153,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL154,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL155,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL156,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL157,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL158,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL159,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL160,ORG-I,RECL-16,KEYL-6,KEYPOS-11
// DISKFILE NAME-FIL161,ORG-I,RECL-16,KEYL-6,KEYPOS-11
-// DISKFILE NAME-FIL162,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL163,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL164,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL165,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL166,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL167,ORG-I,RECL-16,KEYL-6,KEYPOS-11
-// DISKFILE NAME-FIL168,ORG-I,RECL-16,KEYL-6,KEYPOS-11
-// DISKFILE NAME-FIL169,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL170,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL171,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL172,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL173,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL174,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL175,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL176,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL177,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL178,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL179,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL180,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL181,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL182,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL183,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL184,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL185,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL186,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL187,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL188,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL189,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL190,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // DISKFILE NAME-FIL191,ORG-I,RECL-16,KEYL-6,KEYPOS-11
 // SYMFILE NAME-FILS16,DISKFILE-FIL026
 // SYMFILE NAME-FILS17,DISKFILE-FIL027
 // SYMFILE NAME-FILS18,DISKFILE-FIL028
 // SYMFILE NAME-FILS19,DISKFILE-FIL029
 // SYMFILE NAME-FILS20,DISKFILE-FIL030
 // SYMFILE NAME-FILS21,DISKFILE-FIL031
 // SYMFILE NAME-FILS22,DISKFILE-FIL032
 // SYMFILE NAME-FILS23,DISKFILE-FIL033
 // SYMFILE NAME-FILS24,DISKFILE-FIL034
 *
 // PROGRAM NAME-DCP000,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP001,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP002,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP003,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP004,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP005,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP006,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP007,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP008,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
-// PROGRAM NAME-DCP009,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP010,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP011,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
-// PROGRAM NAME-DCP012,MRTMAX-7,DFFMTERM-7,
-//                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP013,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP014,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP015,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP016,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP017,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP018,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP019,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP020,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP021,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP022,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP023,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP024,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP025,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP026,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP027,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP028,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP029,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP030,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP031,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP032,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP033,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP034,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP035,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP036,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP037,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
-// PROGRAM NAME-DCP038,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP039,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP040,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
-// PROGRAM NAME-DCP041,MRTMAX-7,DFFMTERM-7,
-//                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP042,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP043,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP044,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP045,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP046,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP047,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP048,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP049,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP050,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP051,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP052,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP053,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP054,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP055,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP056,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP057,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP058,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP059,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP060,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP061,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP062,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
 // PROGRAM NAME-DCP063,MRTMAX-7,DFFMTERM-7,
 //                     DFFNDF-7,DFFSFDT-258,PACK-F2
```

Figure 18 (Part 4 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// PROGRAM NAME-DCP064,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP065,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP066,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP067,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP068,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP069,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP070,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP071,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP072,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP073,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP074,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP075,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP076,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP077,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP078,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP079,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP080,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP081,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP082,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP083,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP084,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP085,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP086,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP087,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP088,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP089,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP090,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP091,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP092,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP093,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP094,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP095,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP096,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP097,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP098,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP099,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP100,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP101,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP102,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP103,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP104,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP105,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP106,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP107,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP108,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP109,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP110,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP111,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP112,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP113,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP114,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP115,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP116,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP117,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP118,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP119,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP120,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP121,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP122,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP123,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP124,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP125,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP126,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP127,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP128,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP129,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP130,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP131,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP132,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP133,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP134,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP135,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP136,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP137,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP138,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP139,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP140,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP141,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP142,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP143,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP144,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP145,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP146,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP147,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP148,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP149,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP150,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP151,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP152,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP153,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP154,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP155,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP156,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP157,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP158,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP159,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP160,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP161,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP162,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP163,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP164,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP165,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP166,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP167,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP168,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP169,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
```

Figure 18 (Part 5 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// PROGRAM NAME-DCP170,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP171,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP172,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP173,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP174,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP175,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP176,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP177,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP178,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP179,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP180,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP181,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP182,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP183,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP184,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP185,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP186,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP187,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP188,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP189,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP190,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP191,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP192,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP193,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP194,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP195,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP196,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP197,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP198,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP199,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP200,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP201,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP202,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP203,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP204,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP205,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP206,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP207,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP208,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP209,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP210,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP211,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP212,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP213,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP214,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP215,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP216,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP217,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP218,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP219,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP220,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP221,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP222,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP223,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP224,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP225,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP226,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP227,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP228,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP229,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP230,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP231,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP232,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP233,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP234,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP235,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP236,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP237,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP238,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP239,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP240,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP241,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP242,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP243,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP244,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP245,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP246,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP247,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP248,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP249,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP250,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP251,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP252,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP253,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP254,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP255,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP256,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP257,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP258,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP259,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP260,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP261,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP262,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP263,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP264,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP265,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP266,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP267,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP268,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP269,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP270,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP271,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP272,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP273,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP274,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP275,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP276,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP277,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
```

Figure 18 (Part 6 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// PROGRAM NAME-DCP278,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP279,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP280,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP281,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP282,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP283,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP284,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP285,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP286,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP287,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP288,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP289,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP290,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP291,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP292,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP293,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP294,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP295,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP296,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP297,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP298,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP299,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP300,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP301,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP302,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP303,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP304,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP305,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP306,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP307,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP308,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP309,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP310,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP311,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP312,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP313,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP314,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP315,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP316,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP317,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP318,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP319,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP320,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP321,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP322,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP323,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP324,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP325,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP326,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP327,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP328,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP329,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP330,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP331,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP332,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP333,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP334,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP335,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP336,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP337,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP338,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP339,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP340,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP341,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP342,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP343,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP344,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP345,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP346,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP347,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP348,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP349,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP350,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP351,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP352,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP353,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP354,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-F2
// PROGRAM NAME-DCP355,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP356,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP357,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP358,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP359,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP360,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP361,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP362,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP363,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP364,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP365,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP366,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP367,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP368,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP369,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP370,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP371,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP372,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP373,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP374,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP375,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP376,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP377,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP378,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP379,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP380,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP381,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP382,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP383,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
```

Figure 18 (Part 7 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

```
// PROGRAM NAME-DCP384,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP385,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP386,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP387,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP388,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP389,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP390,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP391,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP392,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP393,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP394,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP395,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP396,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP397,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP398,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP399,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP400,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP401,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP402,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP403,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP404,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP405,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP406,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP407,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP408,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP409,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP410,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP411,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP412,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP413,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP414,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP415,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP416,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP417,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP418,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP419,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP420,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP421,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP422,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP423,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP424,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
// PROGRAM NAME-DCP425,MRTMAX-7,DFFMTERM-7,
//                     DFFNDF-7,DFFSFDT-258,PACK-R1
*
// PROGRAM NAME-SML51,FILES-'FILS17/IS,FILS18/IS,FILS16/IS,
// FILS19/IS,FILS20/IS,FILS21/IS,FILS22/IS,FILS23/IS,FILS24/IS',
// PGMDATA-NO
*
// PROGRAM NAME-FL192A,FILES-'FIL000/IS,FIL001/IS,FIL002/IS,
//                           FIL003/IS,FIL004/IS,FIL005/IS,
//                           FIL006/IS,FIL007/IS,FIL008/IS,
//                           FIL009/IS,FIL010/IS,FIL011/IS,
//                           FIL012/IS,FIL013/IS,FIL014/IS,
//                           FIL015/IS,
//                           FIL097/IOU/NOSHR,FIL098/IOU/NOSHR,
//                           FIL099/IOU/NOSHR,FIL100/IOU/NOSHR,
//                           FIL101/IOU/NOSHR,FIL102/IOU/NOSHR,
//                           FIL103/IOU/NOSHR,FIL104/IOU/NOSHR,
//                           FIL105/IOU/NOSHR,FIL106/IOU/NOSHR,
//                           FIL107/IOU/NOSHR,FIL108/IOU/NOSHR,
//                           FIL109/IOU/NOSHR,FIL110/IOU/NOSHR,
//                           FIL111/IOU/NOSHR',PGMDATA-NO
*
// PROGRAM NAME-FL192B,FILES-'FIL016/IS,FIL017/IS,FIL018/IS,
//                           FIL019/IS,FIL020/IS,FIL021/IS,
//                           FIL022/IS,FIL023/IS,FIL024/IS,
//                           FIL025/IS,FIL026/IS,FIL027/IS,
//                           FIL028/IS,FIL029/IS,FIL030/IS,
//                           FIL031/IS,FIL112/IOU/NOSHR,
//                           FIL113/IOU/NOSHR,FIL114/IOU/NOSHR,
//                           FIL115/IOU/NOSHR,FIL116/IOU/NOSHR,
//                           FIL117/IOU/NOSHR,FIL118/IOU/NOSHR,
//                           FIL119/IOU/NOSHR,FIL120/IOU/NOSHR,
//                           FIL121/IOU/NOSHR,FIL122/IOU/NOSHR,
//                           FIL123/IOU/NOSHR,FIL124/IOU/NOSHR,
//                           FIL125/IOU/NOSHR,FIL126/IOU/NOSHR,
//                           FIL127/IOU/NOSHR',PGMDATA-NO
*
// PROGRAM NAME-FL192C,FILES-'FIL032/IS,FIL033/IS,FIL034/IS,
//                           FIL035/IS,FIL036/IS,FIL037/IS,
//                           FIL038/IS,FIL039/IS,FIL040/IS,
//                           FIL041/IS,FIL042/IS,FIL043/IS,
//                           FIL044/IS,FIL045/IS,FIL046/IS,
//                           FIL047/IS,FIL128/IOU/NOSHR,
//                           FIL129/IOU/NOSHR,FIL130/IOU/NOSHR,
//                           FIL131/IOU/NOSHR,FIL132/IOU/NOSHR,
//                           FIL133/IOU/NOSHR,FIL134/IOU/NOSHR,
//                           FIL135/IOU/NOSHR,FIL136/IOU/NOSHR,
//                           FIL137/IOU/NOSHR,FIL138/IOU/NOSHR,
//                           FIL139/IOU/NOSHR,FIL140/IOU/NOSHR,
//                           FIL141/IOU/NOSHR,FIL142/IOU/NOSHR,
//                           FIL143/IOU/NOSHR',PGMDATA-NO
*
// PROGRAM NAME-FL192D,FILES-'FIL048/IS,FIL049/IS,FIL050/IS,
//                           FIL051/IS,FIL052/IS,FIL053/IS,
//                           FIL054/IS,FIL055/IS,FIL056/IS,
//                           FIL057/IS,FIL058/IS,FIL059/IS,
//                           FIL060/IS,FIL061/IS,FIL062/IS,
//                           FIL063/IS,FIL144/IOU/NOSHR,
//                           FIL145/IOU/NOSHR,FIL146/IOU/NOSHR,
//                           FIL147/IOU/NOSHR,FIL148/IOU/NOSHR,
//                           FIL149/IOU/NOSHR,FIL150/IOU/NOSHR,
//                           FIL151/IOU/NOSHR,FIL152/IOU/NOSHR,
//                           FIL153/IOU/NOSHR,FIL154/IOU/NOSHR,
//                           FIL155/IOU/NOSHR,FIL156/IOU/NOSHR,
//                           FIL157/IOU/NOSHR,FIL158/IOU/NOSHR,
//                           FIL159/IOU/NOSHR',PGMDATA-NO
*
// PROGRAM NAME-FL192E,FILES-'FIL064/IS,FIL065/IS,FIL066/IS,
//                           FIL067/IS,FIL068/IS,FIL069/IS,
//                           FIL070/IS,FIL071/IS,FIL072/IS,
//                           FIL073/IS,FIL074/IS,FIL075/IS,
//                           FIL076/IS,FIL077/IS,FIL078/IS,
//                           FIL079/IS,FIL160/IOU/NOSHR,
//                           FIL161/IOU/NOSHR,FIL162/IOU/NOSHR,
//                           FIL163/IOU/NOSHR,FIL164/IOU/NOSHR,
//                           FIL165/IOU/NOSHR,FIL166/IOU/NOSHR,
//                           FIL167/IOU/NOSHR,FIL168/IOU/NOSHR,
//                           FIL169/IOU/NOSHR,FIL170/IOU/NOSHR,
//                           FIL171/IOU/NOSHR,FIL172/IOU/NOSHR,
//                           FIL173/IOU/NOSHR,FIL174/IOU/NOSHR,
//                           FIL175/IOU/NOSHR',PGMDATA-NO
*
// PROGRAM NAME-FL192F,FILES-'FIL080/IS,FIL081/IS,FIL082/IS,
//                           FIL083/IS,FIL084/IS,FIL085/IS,
//                           FIL086/IS,FIL087/IS,FIL088/IS,
//                           FIL089/IS,FIL090/IS,FIL091/IS,
//                           FIL092/IS,FIL093/IS,FIL094/IS,
//                           FIL095/IS,FIL176/IOU/NOSHR,
//                           FIL177/IOU/NOSHR,FIL178/IOU/NOSHR,
//                           FIL179/IOU/NOSHR,FIL180/IOU/NOSHR,
//                           FIL181/IOU/NOSHR,FIL182/IOU/NOSHR,
//                           FIL183/IOU/NOSHR,FIL184/IOU/NOSHR,
//                           FIL185/IOU/NOSHR,FIL186/IOU/NOSHR,
//                           FIL187/IOU/NOSHR,FIL188/IOU/NOSHR,
//                           FIL189/IOU/NOSHR,FIL190/IOU/NOSHR,
//                           FIL191/IOU/NOSHR',PGMDATA-NO
/*
```

Figure 18 (Part 8 of 8). CCP Assignment Set Listing for Example—5704-SC2 Only

DISK STORAGE ESTIMATES FOR THE CCP

Space occupied on the production pack (pack on 'CCUNIT' specified in $EGEN generation control statement).

Object Library:

Base number	1120 sectors	_____
If MLTA supported, add	65 sectors	_____
If BSCA supported, add	45 sectors	_____
If DFF supported, add	65 sectors	_____

Source Library:

Cardless-generation	7 sectors	_____
	Total	_____

Space occupied on a program preparation pack (pack on 'PPUNIT' specified in $EPLG generation control statement):

For LANG-ASSEM, 198 sectors in source library	_____
For LANG-RPGII, 9 sectors in object library	_____
For LANG-COBOL, 2 sectors in object library	_____
For LANG-FORTRAN, 2 sectors in object library	_____
Total	_____

Size of $CCPFILE (determined from specifications in $EFIL generation control statement and generation statements defining lines and terminals):

Determine the number of sectors per assignment set as follows:

Start with one sector	1
For TERMS-n: 1 - 14 terminals, add 3 sectors	
15 - 22 terminals, add 4 sectors	
23 - 28 terminals, add 5 sectors	
29 - 42 terminals, add 6 sectors	
43 - 44 terminals, add 7 sectors	
45 - 51 terminals, add 8 sectors	_____
For BSCA-n ($EBSC) and LINES-n ($EMLA):	
Add one sector if the sum of BSCA-n plus LINES-n is less than seven; add two sectors if the sum of BSCA-n plus LINES-n is seven or greater	_____
If any switched lines are to be supported, add one sector	_____
For DFILES-n, add one sector per 12 or fraction thereof	_____
For PROGS-n, add one sector per 5 or fraction thereof	_____
Total	_____

Determine the space for *all* assignment sets by multiplying the total above by the number specified for SETS-n. To this product add 39 sectors for Program Number 5704-SC1. _____

For Program Number 5704-SC1 only

Determine the number of sectors required by each storage dump.

The supervisor and the CCP partition are written to disk whenever a dump occurs. To calculate the disk space required, 1K bytes of main storage require four sectors of disk space):

1. Add the supervisor size to the CCP partition size.
2. Convert to sectors.
3. Add one sector to this amount; the CCP requires this sector for the header record.

The above calculation results in the number of sectors needed for a dump. _____

For Program Number 5704-SC1 only

Multiply the above number of sectors by the number of dumps specified (DUMPS-n) for the total number of sectors allocated for main storage dump space. +_____

The sum is the total number of sectors allocated for $CCPFILE. _____

For Program Number 5704-SC2 only

The storage dump area calculation for $CCPFILE may be omitted for Program Number 5704-SC2. The storage dumps are placed in the main data area file ($CCPDUMP). This file is used for tracing to disk with the OCC loadable CCP trace; it is also used for storage dumps.

Size of $CCPDUMP (determined from specifications in $EFIL generation control statement)

A. Determine the number of sectors required by each storage dump:
   1. Multiply system size in K by 4. (This yields the number of records per dump.)
   2. Add 1. (One record/dump is required as a header record.)

B. Multiply the above number of records by the number of dumps (DUMPS-n) specified for the total records allocated for the main storage dump space.

C. Convert the above to tracks by dividing by 48 and round up to the nearest number.

D. Add the number of tracks (DPTRAC) designated for use by trace.

E. Add one track (this track will be used as directory space).

Work file space required during generation: Three work files are used, the filenames and the sectors required are:

For $SOURCE, 4080 sectors _____

For $WORK, 1200 sectors _____

For $WORK2, 1200 sectors _____

Total _____

Note: Cardless-generation procedures as supplied require 20 tracks for a $SOURCE workfile at location 386 on the CCP distribution pack.

TIPS FOR IMPROVED PERFORMANCE AND MAIN STORAGE UTILIZATION

This discussion is intended as a summary of techniques for improving CCP performance (response time) and CCP main storage utilization. In some cases, techniques for improving CCP performance result in increased use of main storage. On the other hand, techniques for saving main storage space may result in somewhat slower performance. The proper balancing of these two factors for a particular installation depends on the application requirements of the installation and the hardware resources available to the installation.

Analyzing the application mix of a particular installation will help determine requirements for performance and main storage utilization. In general, applications that call for continuous and prolonged interaction between the terminal operator and the system, such as order-entry applications, require minimum response time so the operator can be as productive as possible. Inquiry applications, on the other hand, might not require such rapid response, because operator interaction with the system is not likely to be continuous, and operator productivity is not likely to be as dependent on the response time of the system. Thus, when the predominant type of application being run under the CCP is order-entry, system design choices can be weighted in favor of improving response time, even if it costs main storage space. When the predominant type of application is inquiry, system design choices can favor saving main storage space.

Resident Accept: The ACCEPT operand of the $EFAC generation statement specifies whether accept-input operations are to be handled by code resident in main storage or by a transient. Resident accept (ACCEPT-YES) reduces transient area contention and provides faster access to data. It is especially effective for single requesting terminal (SRT) program type operations.

Resident DFF Format Index (5704-SC2 only): The DFFINDX operand of the system assignment statement specifies whether the DFF format index built by startup is to be resident in main storage. Resident indexes reduce the time required to locate a DFF format. It is especially effective for single requesting terminal (SRT) programs that use DFF type operations and PRUF type programs.

If DFFINDX-NO is specified, a partial DFF index can be built in main storage if the external pointer list is built and there is room for indexes. Refer to *External Pointer List* for more information.

Resident OPEN/CLOSE: (5704-SC2 only): The RESOPN operand of the $EFAC macro for generation causes $CCROC to be loaded during startup. This eliminates the need to call transients for allocate open and close functions under CCP. In an RPG II program with card input and index load file output, the resident module saves 18 transient calls for one program request. For sort with one input and one output file, the resident module saves 38 transient calls for each program request. However, resident open/close requires 55 bytes in $CC4#1 and 10K bytes of storage to be allocated to the UPA.

Resident Program Request (5704-SC2 only): The RESREQ operand of the $EFAC macro for generation causes $CCRPR to be loaded during startup. This eliminates the need to call transients for CCP program request functions. For an SRT program request or a nonactive MRT program request, four transient loads are saved. For an active MRT program request, three transient loads are saved. Three transient loads are also saved each time the system operator wants a program request (PF9 key). Resident program request also saves two transient loads for each system or terminal operator command that is entered. If RESREQ-YES is specified, $CCRPR occupies 4K bytes of storage, which would otherwise have been allocated to the UPA.

The following system design considerations and techniques can be used to improve CCP performance and/or main storage utilization:

Resident Polling: The RESPOL operand of the *$EBSC generation statement* (see index entry) specifies whether BSC polling modules are to be resident in main storage rather than be executed as transients. Resident polling (RESPOL-YES) is especially important when both DSM partitions are active to ensure good response time. If resident polling is not used, the CCP can be prevented from polling when the non-CCP program partition is using the DSM transient area (for example, for disk access).

Placement of Files and Programs on Disk: CCP performance can be improved through careful arrangement on disk of DSM and CCP routines, $CCPFILE, user application programs, 3270 display formats, and data files. These elements of the system can be arranged in such a way as to keep disk access mechanism movement at a minimum during CCP operation, thus improving disk access speed. Dual spindle systems or the 5445 allow more flexibility in arrangement of these elements than single spindle systems.

DFF Buffer Support (5704-SC2 only): Specify an optional DFF buffer for each BSC line using DFF to improve performance by allowing up to four DFF user PUTs to be scheduled in various stages of transmission at any given time. See *IBM System/3 CCP System Design Guide,* GC21-5165 for additional information.

Minimizing Disk Access Mechanism Contention: The examples shown in Figure 16 for a dedicated system illustrate disk arrangements that minimize disk access mechanism contention. These arrangements would not necessarily be best for all CCP configurations. For example, the location of $CCPFILE is critical to performance when inquiry programs are used frequently or when programs use DFF with many different formats, because program allocation and initiation occur frequently with frequent accesses of $CCPFILE by the CCP. In order-entry applications, on the other hand, program allocation and initiation occur less frequently and the placement of $CCPFILE is less critical; however, placement of data files become critical for good performance, because frequent access to disk data files is likely.

Arranging the elements of the CCP system according to certain general guidelines can improve CCP performance. Some guidelines to follow are:

- Generate CCP on a clean (initialized) disk volume on which an object library was created before generation.
- Place compilers on separate, removable program preparation packs to free object library space that is as close as possible to the beginning of the library on the DSM pack for user programs or 3270 display formats (if used).
- Place user programs and display formats into the object library that has space available nearest the location of the CCP transients ($CC4xx) at the beginning of the library to minimize disk access mechanism movement for retrieval of formats and programs.
- Place $CCPFILE next to the object library on the disk unit that has the smallest object library or as close as possible to the beginning of the disk volume, if no object library is present on that volume. This provides the fastest disk access to $CCPFILE.
- Build packs containing data files evenly, placing the most frequently used files in the center. Then put the remaining data files on each side of the center files according to their usage. In other words, the more often a file is used the closer to center it would be; the less often a file is used the farther from center it would be.

CCP USER TASK PRIORITIES

The priority of user tasks under CCP is determined by the program type. An awareness of the priority structure can help to assure that the relative priority of user tasks meets installation needs. The following list shows the priority structure by program type. FIFO and LIFO refer to first-in-first-out and last-in-first-out respectively. LOW refers to the low priority option on the Assignment PROGRAM statement. This statement is only valid under Program Number 5704-SC2.

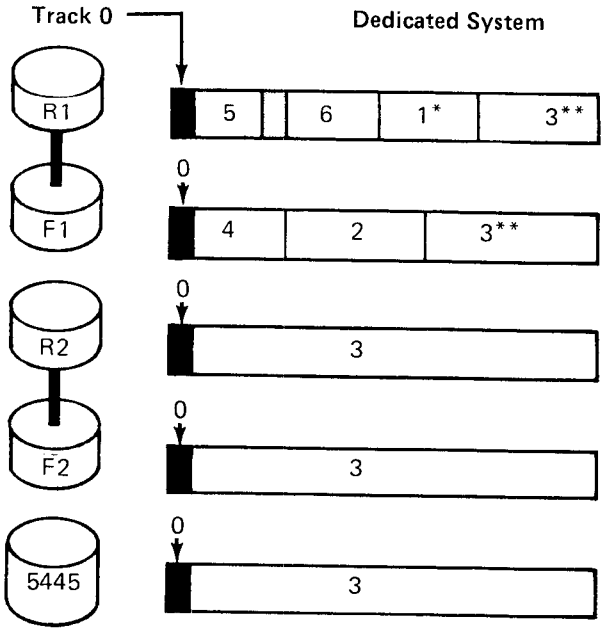
NEP (FIFO)
NEP-LOW (LIFO)
MRT (FIFO)
SRT (LIFO)
MRT and SRT LOW (LIFO)

Interval polling: INTPOL on $EBSC generation, non-MLTA systems, can significantly reduce metered processing unit time in low activity periods.

3284 Printer Busy Timer: If you are using a remote 3284 printer for output and receive numerous printer busy conditions, consider one of the following methods. These methods decrease the amount of line activity required for CCP to process your request for output and to return the code indicating the printer is busy.

1. After receiving a printer busy return code, set the interval timer via the macro $SIT. This stops your task for a specified amount of time before retrying your put operation. Note that this method requires an assembler.
2. Use the RPG II operation code TIME to get the time of day. Issue the time operation until the desired time period elapses from the initial time request. Then reissue the put operation request.
3. Use the CCP operation code WAT to wait for a specified time before retrying the operation.

Note: This method requires a transient to be loaded and can cause conflicts involving the disk drives.



1 = $CCPFILE
2 = User Programs
3 = Data Files
4 = Disk System Management
5 = CCP
6 = 3270 Display Formats (DFF)

CCP Transients ($CC4xx)

Reserved tracks; source library

* $CCPFILE shown in R1 because R1 has a smaller object library in this example.

** Least-used data files.

Figure 16. Examples of Disk Arrangement for Good Access Speed. RESPOL-YES Assumed.

Use of High-Speed Hardware: BSC communication line speed can be increased through use of BSCA high-speed features or EIA local attachment feature (see *IBM System/3 Models 8, 10, 12, and 15 Components Reference Manual*, GA21-9236 for details).

Blocking 3270 data causes minimal degradation in total response time. With blocking, the display format actually begins to appear sooner than without blocking. The screen blinks as it receives each block of data.

Console Activity: Excessive communication with the console by programs running under the CCP can degrade CCP performance by causing a backup of unanswered messages.

Activity of the Non-CCP Program Partition: In dual partition systems, the type of activity in the non-CCP program partition can affect CCP performance. For example, if the program in the non-CCP partition is doing a great deal of disk I/O, contention for disk access can cause slower CCP performance. Also contention for use of the DSM transient area between partitions causes slower CCP performance.

Single Function Inquiry Coding Technique: By entering data to inquiry programs with the program request, the inquiry program needs to perform only two communications operations—an accept input from the terminal and a put to the terminal. If the number of disk accesses done by the program is kept to a minimum, response time for such inquiry programs is minimal and response time for other programs currently in storage is improved.

Updating Shared Files: When two or more programs are updating shared files concurrently, CCP performance can be impaired by two or more programs attempting to update records from the same disk sector at the same time. For example, when a program retrieves a record but takes no action, that record is not updated and the next record is not retrieved.

Using Get instead of Invite and Accept Input: In suitable applications, such as simple inquiry without 3270 DFF, use of get instead of invite input saves one data move operation, saves CCP task switching, and saves the accept input operation.

Using DFF Output/Input Field Types 7 and 8 to Identify Screens: Output/input field types 7 and 8 are protected, alphameric, nondetectable fields with the modified data tag on. Type 7 fields are nondisplay; type 8 fields are displayed at normal intensity. See *IBM System/3 Communications Control Program Programmer's Reference Manual*, GC21-7579 for complete information concerning DFF field types.

DFF output/input field types 7 and 8 are protected and cannot be changed by the terminal operator. These two field types can be overridden to types 1, 2, or 5.

DFF output/input field types 7 and 8 can be used to store information at the terminal. This information is retrieved with the next get or accept input operation as long as the modified data tags are not set off by the program. These field types can be used (by placing information in them with a put message or put override operation) to determine two different things:

1. If the program uses multiple formats to perform its job, these fields could be used to determine which display format the terminal currently is using.

2. If the program has a series of multiple steps or operations to perform with one format, these field types could be used to tell the program which step was performed last with this terminal or which step should be performed next.

The put override operation with reset modified data tags (select input fields) should not be used on a format that contains type 7 or 8 fields because these modified data tags are set off and the data is not returned to the program unless they are overridden to a type 1 or 2. If the type 7 or 8 fields are not overridden to a type 1 or 2, after a reset modified data tags operation, the data at the terminal cannot be changed by the terminal operator because the fields are protected. Because the modified data tags are off, the data is not returned to the program.

The output/input field types 7 and 8 should be defined in the formats such that the data received from these fields would appear immediately after the attention identification character. Recognition of this data and how the following data should be read is then determined.

When the DFF output/input field types 7 and 8 are used to determine which format is presently at the terminal, the accept input or get operation maximum expected length placed in the parameter list (or array) should reflect the length of the longest record area expected from any of the possible formats. Otherwise a *truncated data* return code might be received.

If a single display format is used for all operations, the program can check the value of the output/input field when input is received from the terminal to determine which step in the series to perform. When the program has completed that step, it can issue a put override operation to change the value of the output/input field to call for the next step in the series.

Code All Display Formatting in the Program: In inquiry applications that are highly response-time dependent, performance can be improved by coding all display formatting in the application program rather than using DFF. Response time can be shortened because accessing of display formats on disk is avoided.

Program Request Under Format (PRUF): A program is a PRUF program when the PRUFLNG parameter is specified in the PROGRAM statement. The PRUFLNG parameter specifies the maximum length of program request data acceptable by the program. If the PRUF program is also a DFF program, the PRUF$Z parameter is specified. This gives the name of the format which is used by DFF to format the program request data.

With a non-PRUF request, the maximum amount of data that can be passed to a user program, as a program request, is 78 characters. This causes inefficient usage of the 3270 terminal buffer for program to program communication.

PRUF provides the following capabilities:

- More than one field of data can be passed as program request data.
- More than 78 characters of data can be accepted as program request data.
- The AID character is passed as program request data to PRUF programs but not to non-PRUF programs.
- DFF moves program request data to the user program being requested under format control if the program being requested is a DFF PRUF program. DFF does not process non-PRUF program request data.
- Main storage can be used efficiently because a program does not need to be in main storage during a lengthy terminal operator keying operation.

To inform CCP that the next program request from a 3270 terminal will be a PRUF program request, user program A (which may be a PRUF or a non-PRUF program) executes a PRUF-PUT operation to the 3270 terminal as its last output operation prior to releasing that terminal or going to end-of-job. Before returning the terminal to command mode status, CCP reserves an area from the TP buffer, equal in length to the maximum PRUFLNG, as a temporary hold area for the program request data from that terminal. CCP only reserves a TP buffer area equal in length to the PGMREQL, as specified in the SYSTEM statement, if the last user output operation to that terminal is not a PRUF-PUT operation. The first field entered from the 3270 must be the program name of the PRUF program to be requested and must not begin before row one, column two on the screen. This field is normally output as the first field by program A's PRUF-PUT operation. The terminal operator then keys in data to all needed input fields on the screen.

When all the needed fields are keyed in, press the ENTER key, a PF key, or insert a card into the card reader. (This action is device dependent.) Now program request for program B enters the system. If program B is a non-DFF program, the following data is passed as program request data to program B:

```
CDACCS
CUEI@@B@@PGMNAM¢PGMDATA
 UVD  A
```

Where:

CU = Control unit address of the 3270 terminal

DEV = Device address of the 3270 terminal

AID = Aid character

CC@@ = Cursor address

SBA = Set buffer address (X'11')

@@ = Address of start of PGMNAM field

PGMNAM = Name of program B

¢ = EBCDIC character for a blank (X'40')

PGMDATA = Remainder of 3270 text stream or the number of characters specified by PRUFLNG parameter, whichever is the smaller of the two

A PRUF program request returns more characters of data (eight + PGMNAM length + one, more) than a non-PRUF program. If program B is a DFF PRUF program, DFF attaches the PRUF$Z format to the program. Using that format for control moves data into program B's input record area at program request time.

The following considerations apply when running CCP assignment sets with PRUF programs:

- PRUF PUT operations to the system console are invalid.
- If PRUF is active on the 3270 terminal at program request time, and the program being requested is a non-PRUF program, CCP rejects the program request.
- If PRUF is not active on the 3270 terminal and the program being requested is a PRUF program, CCP issues a 02 return code following the accept input operation. The program request data returned in this case begins with the first character of data following the program name and a blank. This will not have been processed by DFF.
- If PRUF is active on the terminal, all system messages to that terminal are output in positions 82 through 160. Therefore, these positions should be passed with caution at program request time to PRUF programs.
- If PRUF is active on the terminal and the operator presses the CLEAR key, PRUF is inactivated.

The installation verification program (CCPIVP) is furnished as a load (O) module on the CCP PID pack. CCPIVP is a single requester program allowing data with the program request. The program communicates with the 3277 CRT after being requested.

Because of the design of the program, it can be requested from a command terminal as well as the console keyboard. However, once CCPIVP is loaded, it communicates solely through the CRT. If CCPIVP is requested by a terminal rather than the console, the requester is released immediately.

Functions that can be exercised by CCPIVP include:

1. Load from the system console keyboard.
2. Release of a requesting terminal (other than the system console).
3. Data with the request.
4. Single requester programs.
5. Program queuing.
6. Program request resource allocation.
7. Symbolic files including the file specification command, if specified via assignment.
8. Use of put wait, put then get, and put no wait to the console.
9. Allocation, open, close of a 5444 consecutive file under the CCP. Based upon input from the CRT:
   a. CGIVFILE is allocated, opened as a consecutive output file on the 5444.
   b. Records are written to the file.
   c. CGIVFILE is closed.
   d. CGIVFILE is opened as an input file.
   e. CGIVFILE is dumped to the console or to the printer.
   f. CGIVFILE is closed.
10. On unexpected return codes (non-00 for puts, non-00 or non-01 on put then gets), the ability to retry the operation.
11. Concurrent execution of more than one copy of CCPIVP provided symbolic files are used.
12. Use of the console to enter operator commands or to communicate with the program.
13. Closing of CGIVFILE at shutdown.

Two sample assignment sets are supplied, $CGSET and $CGSST. Both include the PROGRAM and DISKFILE statements necessary to run CCPIVP.

Sample assignment set $CGSET is punched into cards during a card-oriented generation. It includes OCL and control statements for running the assignment program, optional link edit OCL, and OCL for CCP startup.

Sample assignment set $CGSST is printed and copied to the users production pack during a cardless generation. $CGSST contains only the assignment control statements and no OCL statements.

Either sample assignment set can be modified according to the user's configuration.

After generation is completed, an assignment build program must be run including a DISKFILE, and PROGRAM statement for CCPIVP in the assignment set. The CCPIVP load (O) module can be executed under the CCP as it exists.

Loading the CCP to Run CCPIVP

```
// LOAD $CCP,xx
```

// FILE (for each actual file to be used by CCPIVP)
See the assignment check actually used to determine the NAME-name entry.

Note: CCPIVP uses the file name CGIVFILE in its disk DTFs.

```
// RUN
```

Procedure for Requesting CCPIVP

1. /FILE CGIVFILE, actual file name (if symbolic files are used)

2a. To load from the console:

CCPIVPbmm/dd/yy

8 char

2b. To load from a terminal:

CCPIVPbmm/dd/yy

8 char

Operating Instructions with CCPIVP

1. Enter data as prompted from the console messages.

2. Messages are sent to the system operator so he can respond to abnormal situations while running CCPIVP.

Normal Operating Messages

DATA ENTERED WILL BECOME A 5444 DISK RECORD OF THE FORM XXX*MM/DD/YY*NNN TO CLOSE FILE, ENTER /*

PLEASE ENTER 03 CHARS OF DATA OR /*

DISK FILE CLOSED, ENTER P TO PRINT OR NON-P TO DISPLAY FILE ON CONSOLE

The following message is displayed if a program is requested without data:

PLEASE ENTER MM/DD/YY

User Error Messages

DATA ENTERED NOT 03 CHARS LONG OR /* PLEASE TRY AGAIN

If an invalid return code is sent back to CCPIVP, the following message is displayed:

ENTER TA TO RETRY OR USE DUMP SYSTEM COMMAND TO DUMP MAIN STORAGE

International Business Machines Corporation

General Systems Division
4111 Northside Parkway N.W.
P.O. Box 2150
Atlanta, Georgia 30055
(U.S.A. only)

General Business Group/International
44 South Broadway
White Plains, New York 10601
U.S.A.
(International)

IBM System/3 Model 15 CP System Reference (File No. S3-36) Printed in USA GC21-7620-6